Q&A of Transformer、Vision Transformer and Multimodal Trandformer

- **Reference（Partly）**：
  - Vanilla Transformer：Attention Is All You Need
  - Vision Transformer (ViT)：An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
  - Multimodal Trandformer：Multimodal Learning with Transformers: A Survey

关于多头注意力机制对跨模态attention的影响可直接查看3.2 Transformer in MultiModel task

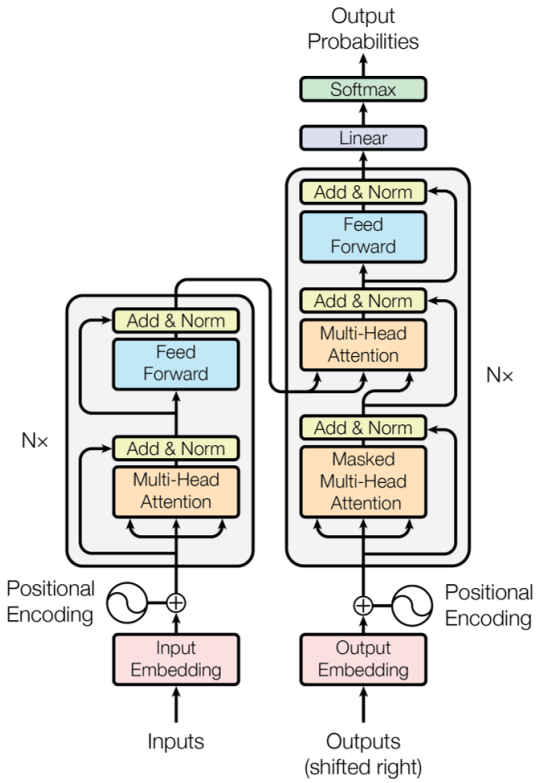
---

1 Vanilla Transformer

1.1 Introduction

- Transformer：一种新的网络架构，完全基于注意力机制，去除了RNN和卷积。可描述input和output之间的**全局依赖关系。**
- RNN的特点(缺点)：无法并行，通常沿着input和output序列的位置计算

1.2 Model Architecture



- **Encoder-Decoder**：x 序列 $\overset{encoder}{\longrightarrow}$ z 序列，z 序列 $\overset{decoder}{\longrightarrow}$ y 序列。需要以自回归（解码时一个个生成）的形式生成output
- **Encoder**：

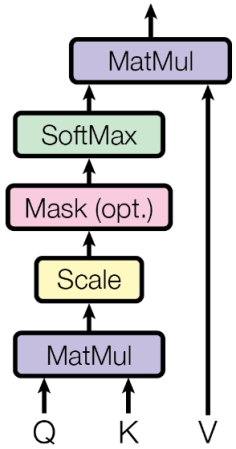
$$Z_1 \leftarrow \mathrm{LN}(\mathrm{MHA}(Z_0) + Z_0)$$
$$Z_2 \leftarrow \mathrm{LN}(\mathrm{FFN}(Z_1) + Z_1)$$

  - 其中 $\mathrm{FFN}(Z_1), Z_1, \mathrm{MHA}(Z_0), Z_0 \in R^{bz \times d_{model}=bz \times 512}$
- **使用 $\mathrm{Layer\ Norm}$ 而不使用 $\mathrm{Batch\ Norm}$ 的原因**：
  - $\mathrm{Batch\ Norm}$：在训练时，对每个feature上的整个mini_batch序列进行归一化，并学到lambda和beta；在测试时，对每个feature上的全部样本进行归一化。
  - $\mathrm{Layer\ Norm}$：对每个样本上的整个feature进行归一化，即针对某个样本内部计算均值和方差。
  - 原因：在CV领域，由于channel维度信息很重要，因此使用BN对每个feature维度归一化可以减少不同feature (channel)信息的损失。在NLP领域，一般序列长度不一致，且各样本的信息关联性不大，因此使用LN无需考虑样本间的依赖，也不需要计算全部样本的均值方差。
- **Decoder**：

$$Z_1 \leftarrow \mathrm{LN}(\mathrm{Masked\_MHA}(Z_0) + Z_0)$$
$$Z_2 \leftarrow \mathrm{LN}(\mathrm{MHA}(Z_1) + Z_1)$$
$$Z_3 \leftarrow \mathrm{LN}(\mathrm{FFN}(Z_2) + Z_2)$$

1.2.1 Attention

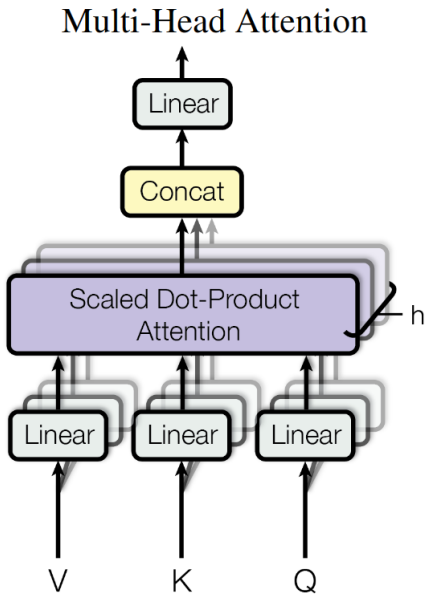
Scaled Dot-Product Attention



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

- **参数维度**：由于 $Q\in R^{N\times d_k}$，$K\in R^{M\times d_k}$，$V\in R^{M\times d_v}$，于是 $\text{Attention}(Q,K,V)\in R^{N\times d_k}$。在 $Encoder$的 MHA 中 $N=M$
- **使用 $dot-product$ 注意力而不使用加法注意力(additive attention)的原因**：通过使用优化的矩阵乘法，更快更省空间。
- **需要 Q 和 K 做点积的原因**：得到的 $\text{softmax}(\frac{QK^T}{\sqrt{d_k}})$ 称为 attention score，视为对 V 进行加权和的权重（来对 V 进行提纯）。
- **Q 和 K 使用不同权重矩阵 W_i^Q,W_i^K 的原因**：在不同空间上的投影，增加表达能力，使计算得到的 attention score 矩阵的泛化能力更高。若拿 $K\cdot K^T$ 进行点积，得到的对称矩阵是投影到了同一个空间，所以泛化能力变差。
- $\frac{1}{\sqrt{d_k}}$ **的原因**：由于 Q,K 中的每个样本 q,k 均已进行 Layer Norm（均值为0，方差为1），则对于 $q\cdot k=\sum_{i=1}^{d_k}q_ik_i$
  - **缩放的原因**：若 d_k 较大，则 $q\cdot k$ 的值会变大，使 softmax 函数趋向于 0 和 1。于是在反向传播时，通过链式法则计算的梯度将会越来越小，最终无法继续训练。为了抵消影响，要对点积的值进行缩放。
  - **缩放参数为 $\frac{1}{\sqrt{d_k}}$ 的原因**：通过计算，$q\cdot k$ 的均值为 0，方差为 d_k。为了使 softmax 更加平滑，需要控制方差为 1，正好需要缩放参数为 $\frac{1}{\sqrt{d_k}}$。

1.2.2 Multi-Head Self-Attention (MHSA)



$$\text{MultiHead}(Q,K,V)=\text{Concat}(\text{head}_1,\dots,\text{head}_h)W^O$$
$$\text{where head}_i=\text{Attention}(QW_i^Q,KW_i^K,VW_i^V)$$

- **参数维度**：由于 $Q\in R^{N\times d_{model}}$，$K\in R^{M\times d_{model}}$，$V\in R^{M\times d_{model}}$；
  而 $W_i^Q\in R^{d_{model}\times d_k}$，$W_i^K\in R^{d_{model}\times d_k}$，$W_i^V\in R^{d_{model}\times d_v}$，$W^O\in R^{hd_v\times d_{model}}$
  于是 $\text{head}_i=\text{Attention}(Q,K,V)\in R^{N\times d_v}$，$\text{Concat}\in R^{N\times hd_v}$，
  $\text{MultiHead}(Q,K,V)\in R^{N\times d_{model}}$。
- **使用多头注意力机制的原因**：$dot-product$注意力没有什么可学习参数，而投影到低维（$d_k=d_v=d_{model}/h=512/8=64$）,投影时的 W,b 均是可学习参数。此外，每次投影可以学到不同子空间的信息，可捕捉多种特征信息。
- **Encoder中的MHA**：这里MHA中的 $query=key=value$ 均来自前一层的输出，即称为self-attention自注意机制。Encoder中的每个位置都可以处理Encoder前一层中的所有(all)位置。
- **Decoder中的Masked MHA1**：这里的Masked MHA1中的 $query=key=value$，即也为self-attention。

$$\text{MSA}(Q,K,V)=\text{softmax}(\frac{QK^T}{\sqrt{d_k}}\odot M)V$$

  - **使用Mask的作用**：为防止Decoder中的信息在 t 时刻看到 t 时刻之后的输入，保证训练和预测时行为一致，并且保持自回归特性（解码时一个一个生成，无法并行化）。通过屏蔽（-1000）softmax 中对应连接后方的值来实现。
- **Decoder中的MHA2**：这里MHA中的 $query$ 来自前面的Decoder，而 key 和 $value$ 来自Encoder的输出，不是自注意力机制。这使得Decoder中的每个位置都可以覆盖输入序列中的所有(all)位置。这模仿了Seq2Seq模型中典型的编码器-解码器注意机制

1.2.3 Feed-Forward Networks (FFN)

$$FFN(x)=\max(0,xW_1+b)W_2+b$$

- **参数维度**：$W_1:[bz,512]\to[bz,2048]$，$W_1:[bz,2048]\to[bz,512]$
- **FFN的作用**：为了语义的转换，先将每个位置的特征向量映射到高维空间，再映射回原始维度。
- **“扩张-压缩”结构的原因**：
  1. 增强特征表达能力：MLP可在高维空间中对特征向量进行更加复杂的非线性变换，从而增强特征的表征能力
  2. 提高模型训练速度：通过压缩，可使模型参数更加紧凑，从而提高模型训练和推理的效率。

1.2.4 Pos Embedding

$$Z \leftarrow X \oplus \text{Pos Embedding}$$

- 为了利用时序信息，加入绝对位置信息pos embedding（这里通过向量加法操作）

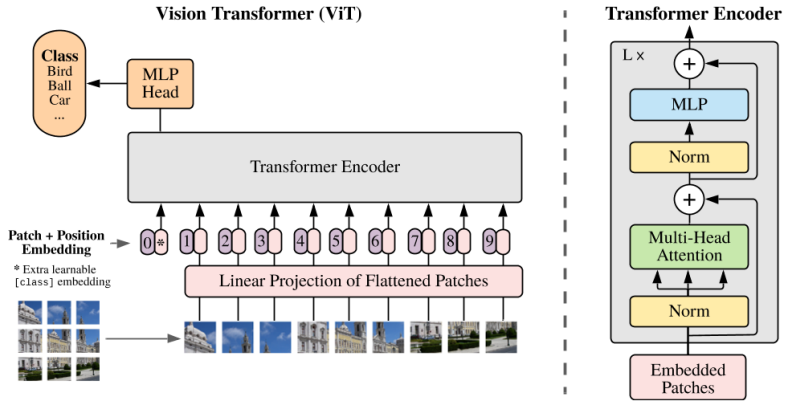
---

2 Vision Transformer (ViT)

2.1 Introduction

- ViT：一种应用于图像patch序列的只用transformer的网络架构，可很好地执行图像分类。
- 主流使用方法：
  1. 先pre-train：在大型数据集上进行预训练
  2. 再fine-turn：在较小的特定任务数据集上进行微调。
- **相同较小数据集下，ViT比ResNet差的原因**：
  - 缺少作为先验信息的归纳偏置(inductive biases)，CNN中两大归纳偏置为locality（靠近的物体相关性更强）和translation equirance（$f \cdot g = g \cdot f$）。
  - 因此ViT需在大规模数据集下进行预训练，来提升效果。实验表明针对大数据集的效果提升未达上限。

2.2 ViT Method



$$
\begin{aligned}
Z_0 &= [X_{class}; X_p^1 E; X_p^2 E; \ldots; X_p^N E] + E_{pos} \\
Z_l' &= \text{MSA}(\text{LN}(Z_{l-1})) + Z_{l-1} \\
Z_l &= \text{MLP}(\text{LN}(Z_l')) + Z_l' \\
y &= \text{LN}(Z_L^0)
\end{aligned}
$$

- **参数维度**：图像 $H = W = 224, C = 3$，**patch**大小 $P = 16(/14/32)$，patch个数 $N = HW/P^2 = 196$

  $X_p^i \in R^{N \times (P^2 \cdot C) = 196 \times 768}$，$E \in R^{768 \times 768}$：$[bz, 196, 768] \rightarrow [bz, 196, 768]$，

  $X_{class} \in R^{1 \times 768}$，$E_{pos} \in R^{197 \times 768}$

  $Z_i \in R^{197 \times 768}$

- **使用步骤**：将图像分割为多个patches，将这些patches的线性投影序列加入cls token和pos embedding后，作为Transformer Encoder的输入。最后取class token的特征进行分类。训练时以监督学习的方式进行图像分类。

2.2.1 Embedding

- **将image分割为多个patch的原因**：传统CNN中，可有效捕捉图像的局部特征。但在Transformer中每个token都关注的是全局信息，因此需要将全局分割成更小的部分，以便Transformer可以更好地处理局部信息。此外Vanilla Transformer适用于NLP，其输入token为2维（不算batchsize），而传统RGB图像输入为3维，因此需要将输入变为2维，故在划分为patch后实现flatten操作进行降维。
- **class token的作用**：class token经过Transformer Encoder后，通过全局注意力，学到所有patch中的权重。然后将代表所有token信息的class token作为整个Transformer的**输出特征**，放入分类器MLP Head。
    - 不使用class token：类似ResNet的方法，将patch token在Transformer Encoder的输出作为图像特征放入全局平均池化(GAP)，通过调整learning rate，效果类似。
- **position embedding**：可视作一种额外的输入特征。
    - **作用**：为了区分处理seq中不同位置的信息，从而更好地关注序列中的局部结构。具体的说，为input中的每个位置提供一些时间或空间信息，并将其映射到一个理论上的特征空间坐标上。
    - **方法**：一维pos embedding，二维pos embedding，相对pos embedding

2.2.2 Transformer Encoder

> 类似Vanilla Transformer中的Encoder

2.2.3 MLP Head

- **分类头将class token的特征作为输入**：在pre-train时，由一个带hidden layer的MLP实现。在fine-turn时，由一个线性层实现。

2.3 Fine-tuning

- 最后的分类器可直接替换为输出 $\mathrm{num\_class}$ 维的线性层
- **图像尺寸与pre-train时不同时**：由于pre-train的尺寸无法与迁移的下游数据集对应，因此pos-embedding无法直接使用。可通过**二维插值**实现。

2.4 Comparision

- ViT和ResNet比较：大规模数据集训练时（ImageNet21K及更大数据集），ViT效果更好

---

3 MultiModel Transformer

3.1 Transformer understand（理论回顾）

3.1.1 Tokenized Input

- **Tokenized Input的作用**：通过标记化序列(tokenized sequences) 实现**可将任何模态输入建模为全连通图。**
- **Tokenized Input的优势**：
    1. 缓解跨模态引起的模态壁垒
    2. 通过concat,stack和weighted sum等多种方式处理输入信息
    3. 与特定token（例：[class], [MASK]）兼容
    4. 帮助attention处理多模态数据。

3.1.2 Attention Mechanism

- **Self-Attention的作用**：使输入序列的每个元素关注所有其他元素，因此可对self-attention视为将输入编码为一个全连通图。
- **Multi-Head Attention的作用**：相当于一种帮助模型并行处理多个表征空间的信息的整体思想。
- **Masked Self-Attention的作用**：相当于向Transformer模型加入对应领域的某种先验知识。

3.2 Transformer in MultiModel task

- **多模态学习的目的**：如何使不同模态的信息更好的align或fuse。
    - 双塔模型的目的：主要关注多模态的**align。**适用于多模态检索、匹配的任务，例如CLIP。
    - 单塔模型（充分利用**Cross-Attention**的架构优势）的目的：主要关注多模态的**fuse。**适用于VQA、Image Caption等信息融合、推理的任务，例如ViLT，ViLBERT。

    > 于是，Cross-Attention的目的是：如何从不同模态中更好的提取重要的特征，并进行融合。

- **用Transformer做多模态学习的优点**：
    - **任务方面**：
        - 对比过去方法：过去的多模态主要方法是CNN提取图像特征，RNN提取文本特征等等，对特征做融合后再经过分类器。这种模型大量依赖不同模型的输出特征，来进行多重操作，pipeline太过复杂。
        - Transformer的优势：Transformer可对不同模态使用同一个模型，在应用时直接将多种模态的token输入进模型，可省去特征提取步骤，便于实现end2end的网络构架。因此使模型能够在新的多模态数据上进行良好的推广，从而提高**泛化能力。**
    - **原理方面**：
        - 对比过去的方法：CNN只能提取**局部矩阵的特征**，再汇聚到一起。RNN虽然可处理更长的输入，但结构问题使得每个状态包含上一个状态的输出，在反向传播的链式求导时会产生**长程依赖**，忘记靠前的输入（LSTM的长程记忆功能是一种解决方法）。
        - Transformer的优势：
            - **Tokenized Input**：作为输入步骤已经完成完成了一部分Early Fusion，可简化Attention对跨模态特征的相关性的学习过程。
            - **Multi-Head Attention（多头注意力机制对跨模态attention的影响）**：
                1. **并行能力角度**：可以通过矩阵乘法来提高处理全局特征的并行能力。
                2. **全局特征角度**：通过Multi-Head可提取全局数据在不同空间的特征，进行融合。还可防止输入过长带来的遗忘问题。
                3. **注意力角度**：可以计算每个token相对其他token（可能是不同模态的token）的相似度，作为attention score来计算对全局的贡献程度（例如：确定一张图

像的某些重要区域，与文本描述相匹配；确定视频中的某些重要帧，作为视频的特征表示来提高检测性能）。

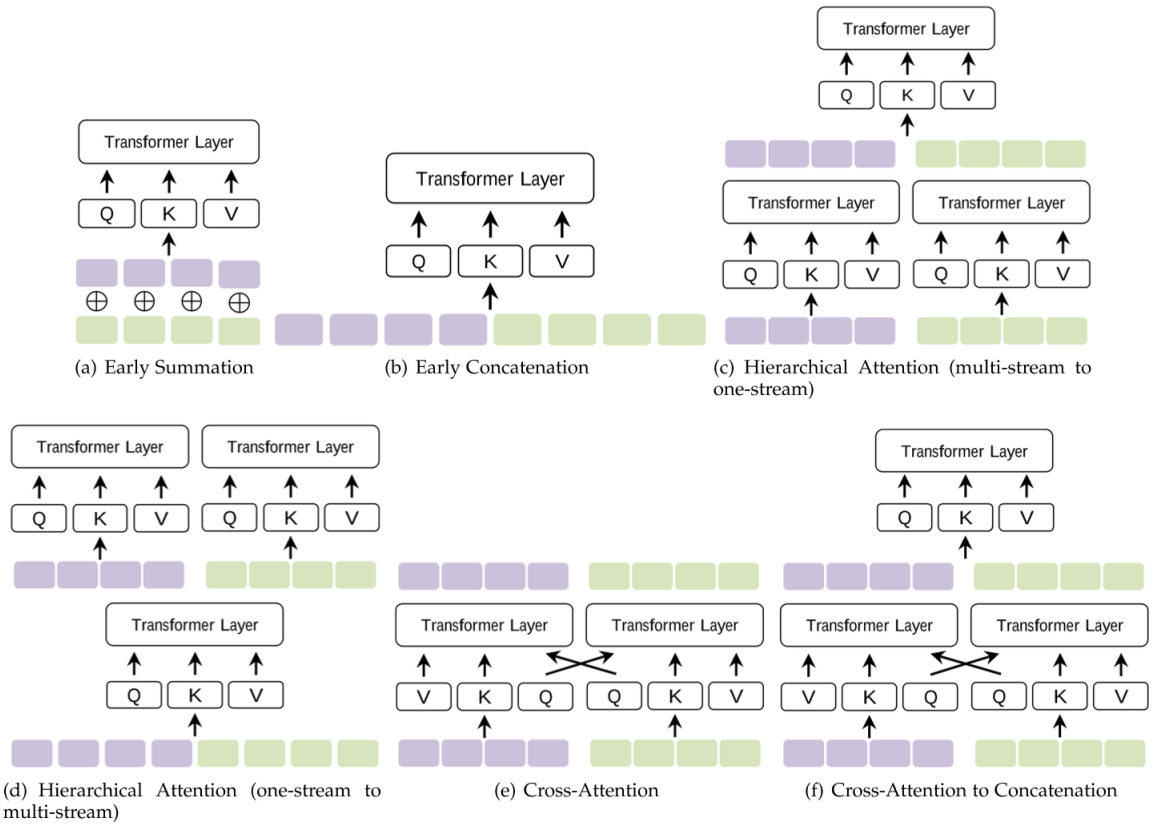
4. **数据偏差（鲁棒性）角度**：作为attention机制，通过对不同模态的特征进行加权，使模型更关注那些对任务重要的特征，从而减少特定模态的数据偏差（例如：图像的拍摄角度、光照；文本的语言习惯、文化背景）对模型的影响，并缓解多模态数据太过丰富带来的噪声和干扰问题，提高模型的鲁棒性。
5. **几何拓扑角度**：Attention可以看作将不同模态的输入建模为全连接图（读取全局信息），将每个输入token视为图中的节点，这样可对不同模态数据选择合适的建模空间。

- qkv和multi-head结构在align方面的负面影响：
  -

3.2.1 MultiModel Input

- 可在输入层面**将embedding后的token视为节点，将整个input构造成全连接图**。这种通用的结构使Transformer可在不同模态下有效。此处的融合处理可视为一种**early fusion**。
- **Special Token**：在语义上视为token序列的占位符（例：[CLS]，[SEP]）
- 处理任意模态的输入，只需执行两个主要步骤：
  1. 令牌化(tokenize)输入
  2. 选择一个embedding space来表示tokens，然后将数据输入到Transformer。

3.2.2 Self-Attention Variants



1. **Early Sum(token-wise sum)**：在token input时进行加权和实现。
2. **Early Concat**：
   1. 缺点：由于attention的复杂度为 $O(n^2)$，因此直接concat会增加复杂度
3. **Hierarchical Attention (multi-stream to one-stream)**
4. **Hierarchical Attention (one-stream to multi-stream)**
5. **Cross-Attention**：将一模态的query交换到另一模态的attention中，实现跨模态交互

$$\text{softmax}(k \cdot (S_2 W_Q)(S_1 W_K)^T) S_1 W_V$$

- 优点：复杂度并没有增加
- 缺点：虽然学习了跨模态信息，但并没有对自己模态的上下文信息进行self-attention

6. **Cross-Attention to Concat**
   - 优点：可缓解Cross-Attention不能学习自己模态的缺点。