Attention注意力机制

Self-Attention

输入可能是输入也可能是某一隐藏层的输出，最后的每一个输出都是考虑所有输入。
产生向量对应输出的时候首先考虑所有输入中与该向量相关的输入
自注意力机制是考虑的时间序列内部的依赖

计算Q(query)，K(key)，V(value)

对于输入矩阵$X \in \mathbb{R}^{\text{batch_size} \times \text{seq_len} \times d_{\text{model}}}$，生成查询向量`Q`，键向量`K`和值向量`V`。这是通过乘以权重矩阵得到的。
$Q = X * W^Q$
$K = X * W^K$
$V = X * W^V$
其中，W^Q, W^K, W^V是训练过程中学到的权重矩阵，W^Q, W^K的维度是$\mathbb{R}^{d_{model} \times d_k}$，$W^V$的维度是$\mathbb{R}^{d_{model} \times d_v}$

`Q`，`K`的形状是$\mathbb{R}^{\text{batch_size} \times \text{seq_len} \times d_k}$，`V`的形状是$\mathbb{R}^{\text{batch_size} \times \text{seq_len} \times d_k}$，其中`d_k`是输入$X$经过线性变换后的维度
d_{model}在文本处理中表示输入词编码向量的维度，其中词编码包含索引编码和位置编码，每一个输入的词就是一个经过编码后的向量，输入的X就是由`seq_len`个单词输入，`batch_size`相当于多个句子

计算相关性

计算查询向量`Q`与所有输入(包含自己)对应键向量`K`之间的相似性，一般来说是通过点积(Dot Product)来实现的

$$score = Q * K^T$$

得到每个`batch`的相似性矩阵维度为$\mathbb{R}^{seq_len \times seq_len}$
计算过程中每个向量的Q都会跟每个向量的K做矩阵乘法运算，对于向量本身来说肯定是向量自己与自己的点积最大

然后为了避免点积的值过大，导致`softmax`的值偏向两端靠拢，通常会对结果进行缩放，具体来说就是将点积的结果除以一个缩放因子$\sqrt{d_k}$，其中d_k是查询向量的维度

$$score = \frac{score}{\sqrt{d_k}}$$

最后输出的时候就是进行一个`softmax`

$$score' = softmax(score)$$

得到表示的是两个输入之间的相关性

mask机制

对于文本建模来说一般都是依照前面的信息来对后续的文本进行预测，因此不能让模型看到后面文本的信息。
例如在`[The, cat, sat]`中，我们希望"The" 只能看自己，"cat" 可以看 "The" 和自己，"sat" 可以看 "The"、"cat" 和自己。
因此引入一个mask矩阵

$$\text{Mask} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

在实际实现中上三角矩阵中上三角部分被定义为负无穷，下三角(包含对角线)部分被定义为0，最后进行一个softmax后上三角矩阵就变成0，被完全忽略
$score' = Mask * score'$
掩码机制实际上就是为了实现自回归功能

然后根据$\alpha_{i,j}$抽取`V`中重要的信息最后得到输出

$$output = scroe' \times V$$

综上所述，`Attention`的数学公式可以概括为：

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

Multi-head Self-Attention

为什么引入多头注意力机制：
单个注意力头可能只关注某一特定类型的关联信息，对于复杂数据无法全面进行描述，因此引入多个注意力头的多头注意力机制

既然多头注意力机制的输入都是一样的，如何保证多个注意力头关注的是不同的信息？
权重随机初始化可以使得训练的时候权重会朝着不同的方向进行优化，从而关注不同的信息

在原有的基础上引入多个QKV
有以下的公式：
$Q_h = X * W_h^Q$
$K_h = X * W_h^K$
$V_h = X * W_h^V$
其中`h`表示`head`的个数
最后的操作跟之前一样最后得到$output_h$
最后将多头输出接起来乘以一个权重矩阵W^O得到$output$

$$MultiHead(Q,K,V)=Concat(head_1,\dots,head_h)W^O$$
$$where\ head_i=Attention(XW_i^Q,XW_i^K,XW_i^V)\ \ i=1,2,\dots h$$

Cross-Attention

在交叉注意力中`Q`和`K/V`来自不同的输入序列或者不同的模态
交叉注意力机制关注的是两个序列之间的关联

交叉注意力使用两个序列，源序列`X`经过运算后得到`K,V`，目标序列`Y`经过运算得到`Q`

实际上交叉注意力做的事情就是一个语义对齐操作