# 理论分析

## 主要特点

传统的Transformer输入是限定长度的句子，一般在100~200左右，使得其无法处理太长的句子，也无法生成长文本。

一种改进模型是vanilla Transformer，它将一个长句子切分成若干小段(segment)，如图1.1，具体怎么处理暂不清楚。

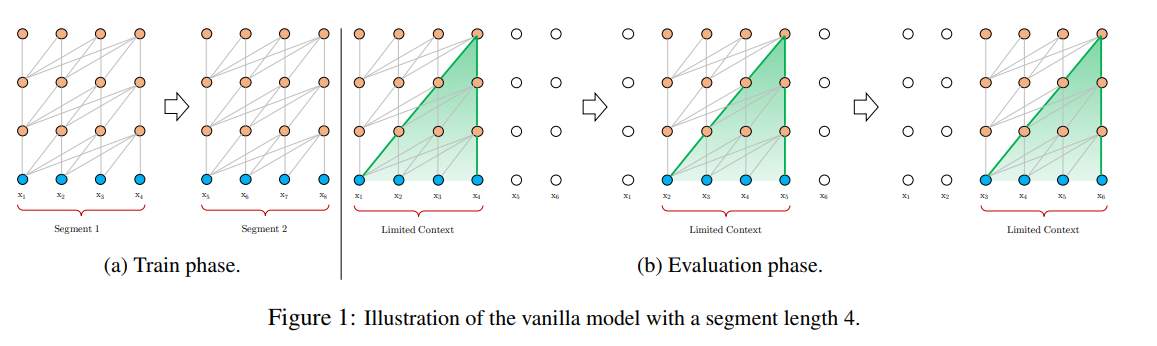


图 1.1 vanilla Transformer

这种方式的缺点就是上下文碎片化，每个segment都是单独计算的，没有考虑到segment之间的上下文联系。Transformer-XL在其基础上进行了改进，在计算下个segment时将上个segment产生的中间编码当作额外的输入用于产生K、V。

Transformer-XL的主要特点如下：

1. 使用循环深度自注意力模型结构，将长句子划分成多段进行处理
2. 考虑到段与段之间的联系，复用之前段提取到的信息
3. 使用相对位置编码而不是绝对位置编码，并提出了简单有效的相对位置编码计算公式

## 计算方式

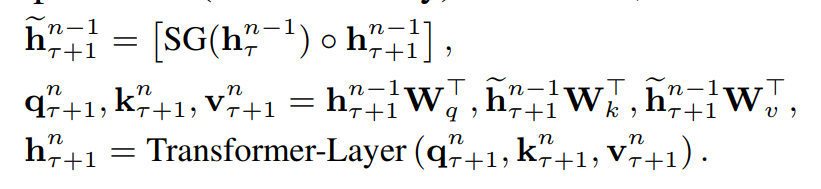
定义：

输入的完整句子：

第个segment： ，segment长度为L

第个segment在第n层Transformer产生的中间编码：

计算方式：



1. SG代表梯度停止，即在计算时，不再对之前的相关参数进行优化？
2. Q还是只通过生成，K、V则直接使用了上个segment的中间编码

源码实现：

1. 先将和直接在第一个维度上cat在一起，生成新的
2. 用h经过全连接层生成Q、K、V
3. Q再去掉前半部分，只留下相关的后半部分
4. 因此K、V的第一维度和Q是不同的

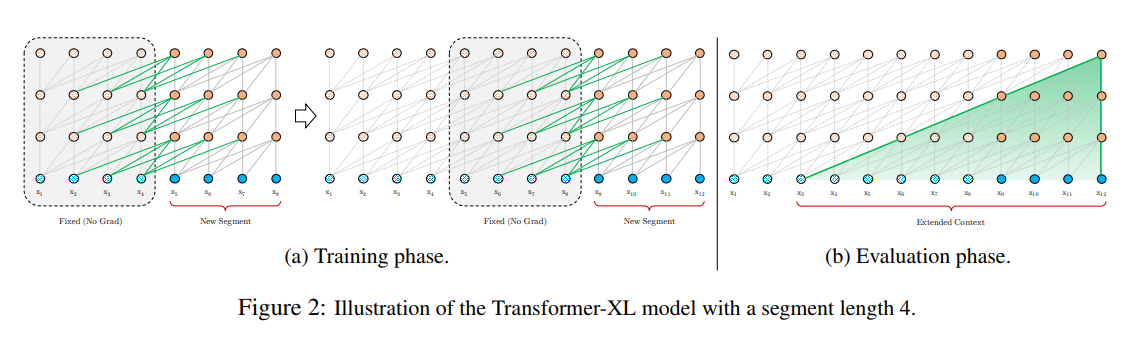


图 1.2 Transformer-XL

当segment长度为4时，每个的隐藏层在计算时只能用4个，比如

实现细节：

1. 每次计算时，mult-attention输入的是上个segment的中间输出mem与当前segment的embedding/中间输出

## 相对位置编码

为了在实现上述想法时保留合理的位置编码信息，本文使用相对位置编码来计算Attention score。

对于与，传统Transformer中Attention score的计算方式如图1.3所示，其中E代表相应的词向量，U代表相应的绝对位置编码。

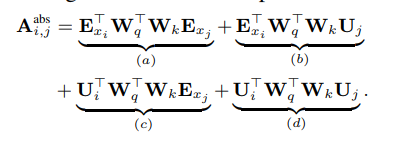


图 1.3 使用绝对位置编码计算score

相对位置编码的改进有两点：

1. 使用相对位置编码代替，也是通过三角函数计算出的，不需要学习
2. 使用训练得到的矩阵代替

新公式如图1.4所示。

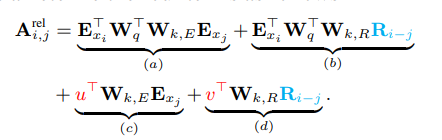


图 1.4使用相对位置编码计算score