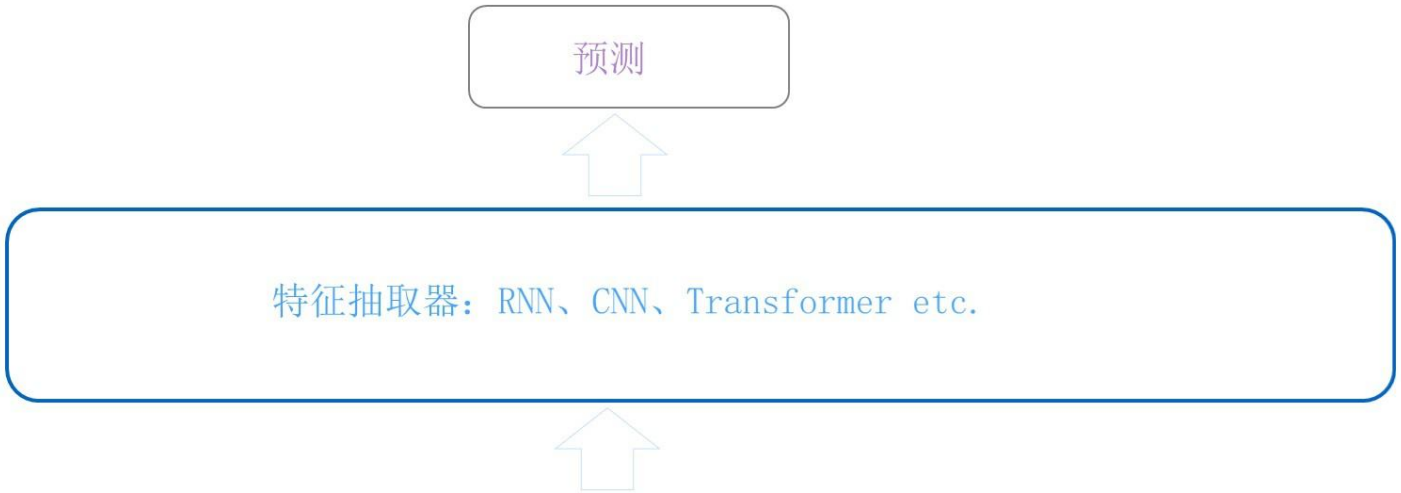
Attention模型中序列的方向信息和位置信息编码

汇报人：吴林志

NLP任务的特点



如果你愿意一层一层的剥开我的心……那么你会坐牢的我跟你说。

NLP任务的特点和图像有极大的不同

NLP的输入往往是一句话或者一篇文章，所以它有几个特点：

- 输入是个一维线性序列
- 输入是不定长的，有的长有的短
- 序列的**方向信息**很重要：　例：这件衣服贵的很啊！　（“很”是修饰“贵”的）
- 单词或者子句的**相对位置关系**很重要，两个单词位置互换可能导致完全不同的意思
  - 例：“你欠我那一千万不用还了”和“我欠你那一千万不用还了”
- 句子中的**长距离特征**对于理解语义也非常关键
  - 例：“如果……，那么……”。预测“那么”时依赖于“如果”。

https://arxiv.org/pdf/1706.03762.pdf

Self-Attention

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

subspace representation

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

$$att_{additive}(h_j, h_i) = V_a \tanh(U_a h_j + W_a h_i + b_a)$$

$$att_{bilinear}(h_j, h_i) = h_j^T W_{bil} h_i$$

$$att_{multiplicative}(h_j, h_i) = h_j^T h_i$$

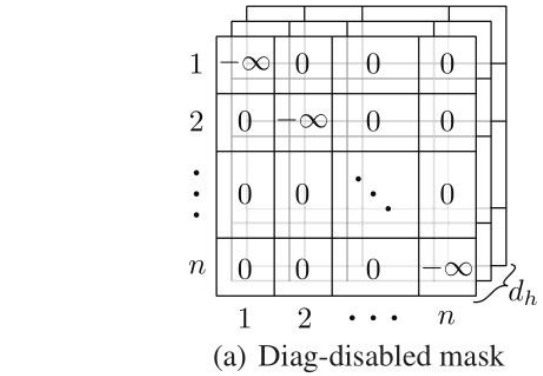
$$a(h_j, h_i) = \frac{\exp(att(h_j, h_i))}{\sum_{\tilde{i}=0}^{N} \exp(att(h_j, h_{\tilde{i}}))}$$

$$h_j^* = \sum_{i=0}^{N} a(h_j, h_i)\, h_i$$

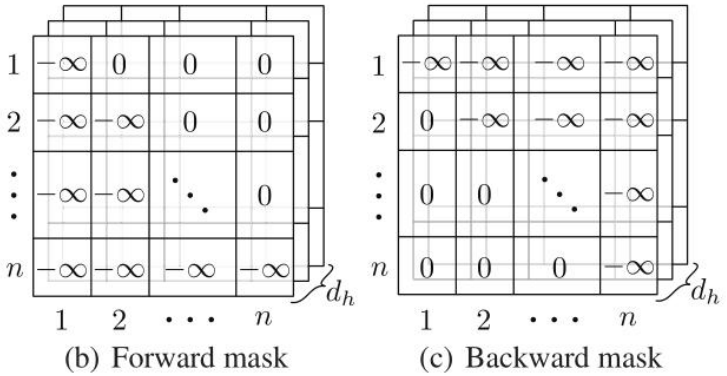
如何编码Self-Attention中的方向信息?

https://arxiv.org/pdf/1709.04696.pdf

Directional Self-Attention Network (DiSAN)



(a) Diag-disabled mask



(b) Forward mask (c) Backward mask

Figure 3: Three positional masks: (a) is the diag-disabled mask M^{diag}; (b) and (c) are forward mask M^{fw} and backward mask M^{bw}, respectively.

$$M_{ij}^{diag} = \begin{cases} 0, & i \neq j \\ -\infty, & i = j \end{cases}$$

disable the attention of each token to itself

$$M_{ij}^{fw} = \begin{cases} 0, & i < j \\ -\infty, & otherwise \end{cases}$$

the only attention of later token j to early token i

$$M_{ij}^{bw} = \begin{cases} 0, & i > j \\ -\infty, & otherwise \end{cases}$$

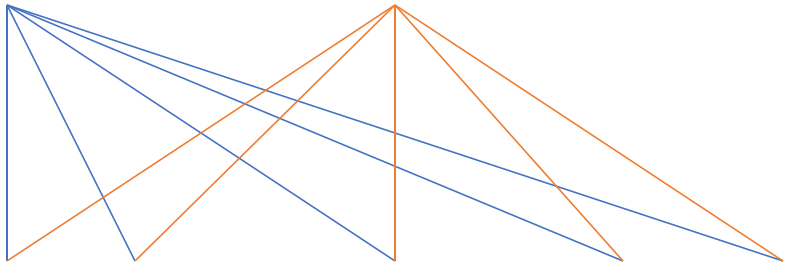
the only attention of later token i to early token j

$M_{ij} = -\infty \iff$ there is no attention of x_j to x_i

$M_{ji} = 0 \iff$ attention of x_i to x_j exists

举个栗子：

I come from China !

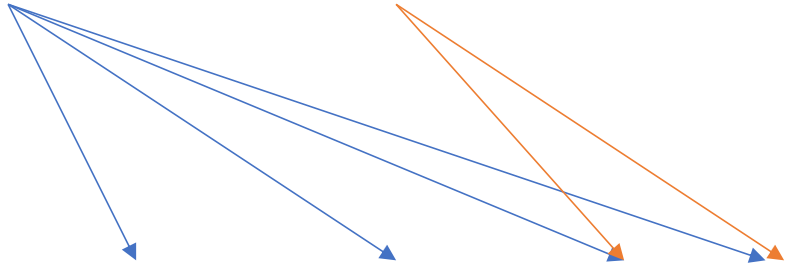


I come from China !

undirectional

forward

I come from China !

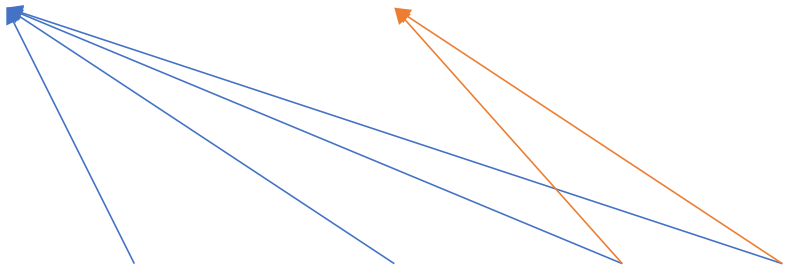


I come from China !

---

backward

I come from China !



I come from China !

如何编码Self-Attention中的位置信息?

RNN和Attention位置编码的差异

- RNN 通常依靠其循环机制，结合t时刻的输入和前一时刻的隐层状态h_{t-1}计算出h_t，直接通过其顺序结构沿时间维度捕获相对和绝对位置。而非 RNN 模型不需要顺序处理输入，则需要显式编码才能引入位置信息。
- 在同一个输入序列中，不同位置的相同单词的representation完全相同，这样并不能体现单词之间的时序关系，所以要对单词的时序位置进行编码表征。

Transformer的位置编码

- 采用绝对位置编码:

$$PE(i,2k) = \sin(\frac{i}{10000^{2k/d\_model}})$$

$$PE(i,2k+1) = \cos(\frac{i}{10000^{2k/d\_model}})$$

序列:	I	come	from	China	!
位置:	0	1	2	3	4

如何变成相对位置编码?

$$PE_{ij}(2k) = \sin(\frac{j-i}{10000^{2k/d\_model}})$$

$$PE_{ij}(2k+1) = \cos(\frac{j-i}{10000^{2k/d\_model}})$$

序列:	I	come	from	China	!
位置:	-2	-1	0	1	2

https://arxiv.org/pdf/1803.02155.pdf

如何变成相对位置编码?

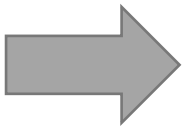
Relative Position Representations

输入: $x = (x_1, \ldots, x_n)$　输出: $z = (z_1, \ldots, z_n) \in \mathbb{R}^{n \times d}$:

$$e_{ij} = \frac{x_i W^Q (x_j W^K)^T}{\sqrt{d_z}}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{n} \exp(e_{ik})}$$

$$z_i = \sum_{j=1}^{n} \alpha_{ij} (x_j W^V)$$



$$e_{ij} = \frac{x_i W^Q (x_j W^K + a_{ij}^K)^T}{\sqrt{d_z}}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{n} \exp(e_{ik})}$$

$$z_i = \sum_{j=1}^{n} \alpha_{ij} (x_j W^V + a_{ij}^V)$$

RPR 的不在输入时将位置表示与 token 表示相加，而是选择对 self-attention 进行改动!

$$a_{ij}^K = w_{\text{clip}(j-i,k)}^K$$

$$a_{ij}^V = w_{\text{clip}(j-i,k)}^V$$

$$\text{clip}(x, k) = \max(-k, \min(k, x))$$

useless

模型学习相对位置表示 $w^K = (w_{-k}^K, \ldots, w_k^K) \in \mathbb{R}^{(2k+1) \times d}$ (w^V 同理)，同一层的attention heads之间共享，但是在不同层之间是不同的!

https://arxiv.org/pdf/1901.02860.pdf

Transformer-XL

非常长距离的Transformer

- ✓ Segment-level recurrence
- ✓ Relative Positional Encoding

片段水平的复现和状态重用
Segment-level recurrence with state reuse

- 为了解决只能利用一个固定长度的context的限制，文中提出了一个recurrence机制。**在训练的时候，先前的segment计算出的隐状态序列，被固定和储存下来，在下一个segment中被当做延长了的context（扩展上下文）再使用**。尽管梯度只留在一个segment里，这个额外的输入使得网络可以利用历史上的信息，使网络向着记住长距离依赖和解决上下文碎裂问题的方向前进。

片段水平的复现和状态重用
Segment-level recurrence with state reuse

- 令两个长度为L的片段分别是 $s_\tau = [x_{\tau,1}, \dots, x_{\tau,L}]$和$s_{\tau+1} = [x_{\tau+1,1}, \dots, x_{\tau+1,L}]$。表示第$\tau$个片段$s_\tau$的第$n$个隐状态序列为$h_r^n \in R^{L\times d}$, d是隐层的维度. 那么片段$s_{\tau+1}$ 的第n个隐状态序列依如下产生

q, k, v: $L\times d, (M+L)\times d, *$

$$\widetilde{\mathbf{h}}_{\tau+1}^{n-1} = [\mathrm{SG}(\mathbf{h}_\tau^{n-1}) \circ \mathbf{h}_{\tau+1}^{n-1}], \quad \text{(extended context)}$$

$$\mathbf{q}_{\tau+1}^n, \mathbf{k}_{\tau+1}^n, \mathbf{v}_{\tau+1}^n = \mathbf{h}_{\tau+1}^{n-1}\mathbf{W}_q^\top, \widetilde{\mathbf{h}}_{\tau+1}^{n-1}\mathbf{W}_k^\top, \widetilde{\mathbf{h}}_{\tau+1}^{n-1}\mathbf{W}_v^\top, \quad \text{(query, key, value vectors)}$$

$$\mathbf{h}_{\tau+1}^n = \text{Transformer-Layer}\left(\mathbf{q}_{\tau+1}^n, \mathbf{k}_{\tau+1}^n, \mathbf{v}_{\tau+1}^n\right). \quad \text{(self-attention + feed-forward)}$$

the function $SG(\cdot)$ stands for stop-gradient, the notion $[h_u \circ h_v]$ indicates the concatenation of two hidden sequences along the length dimension, $W.$ denotes model parameters.

片段长度为4时Transformer-XL的训练和评估过程

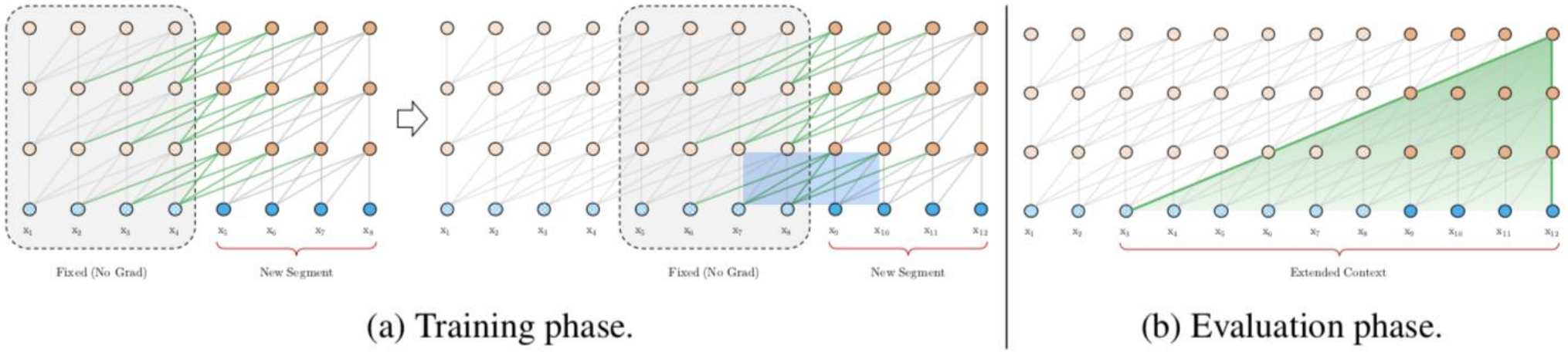


Figure 2: Illustration of the Transformer-XL model with a segment length 4.

相对位置编码 relative positional encodings

- 在标准的Transformer中，序列顺序的信息由位置编码提供，可以用$U \in \mathbb{R}^{L_{max} \times d}$表示，第$i$行对应片段中第$i$个绝对位置，$L_{max}$是模型输入的最大长度。Transformer的真实输入是element-wise的word embedding和位置编码的加和（和segment embeddings）。
- 采用这样的位置编码在文中的复现机制中，模型没有信息来区分先前和当下的片段$(x_{\tau,j}, x_{\tau+1,j})$在位置上的不同。
- 为避免这种失败的模式，基本的想法是在隐状态中编码相对位置信息。概念上讲，位置编码给了模型在如何收集信息上一个时间的提示或者“倾向”，也就是，哪里需要注意。
- 基于同样的目的，除了将这样的倾向静态的合并到开始的嵌入层，也可以将同样的信息引入到每一层的注意力得分上。更重要的是，将时间的倾向定义为一种相对的方式更符合直觉和更泛化。

相对位置编码 relative positional encodings

- 可以定义一组相对位置编码$R \in \mathbb{R}^{(M+L)\times d}$，第$i$行$R_i$表示两个位置的相对距离为$i$时的编码。这样模型就能区分出$(x_{\tau,j}, x_{\tau+1,j})$。
- 在标准Transformer中，在同一个片段中query q_i 和 key 向量 k_j 之间的注意力得分可以分解为：

$$\mathbf{A}^{\text{abs}}_{i,j} = q_i^\top k_j = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)}.$$

$(W_q(E_{xi}+U_i))^T \cdot (W_k(E_{xj}+U_j))$

- 根据只依靠相对位置信息的想法，可以重新用参数表示这四项：

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(b)} + \underbrace{u^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(c)} + \underbrace{v^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(d)}.$$

$(W_q(E_{xi}+U_i))^T(W_k E_{xj}) + (W_q(E_{xi}+U_i))^T(W_k U_j)$ ⇒ $(W_q E_{xi}+u)^T(W_{k,E} E_{xj}) + (W_q E_{xi}+v)^T(W_{k,R} R_{i-j})$

相对位置编码 relative positional encodings

1. 将所有的key vector中的绝对位置编码U_j替换为相对位置编码R_{i-j}。R是一个正弦编码矩阵（attention is all you need），没有可学习的参数。
2. 引入可训练的参数$u \in \mathbb{R}^d$代替term(c)中的query $U_i^\intercal W_q^\intercal$。也就是，对于任何的query位置，query向量都一样，表明不管query中位置如何，对不同单词的注意力倾向是相同的。同样的原因，加入一个可训练的参数$\mathbf{v} \in \mathbb{R}^d$来代替term(d)中的$U_i^\intercal W_q^\intercal$。
3. 分离两个权重矩阵$W_{k,E}$和$W_{k,R}$，分别产生基于内容的key向量和基于位置的key向量。

相对位置编码 relative positional encodings

- 在新的参数下，每一个term都有一个直觉上的意义:
- term(a) 代表**基于内容的寻址**;
- term(b) 捕获**基于内容的位置偏差**;
- term(c) 管理一个**全局的内容偏差**;
- term(d) 编码了**全局的位置偏差**。

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^{\top}\mathbf{W}_q^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^{\top}\mathbf{W}_q^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)} + \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

$\mathbb{R}^{1\times d}\ \mathbb{R}^{d\times d}\ \mathbb{R}^{d\times d}\ \mathbb{R}^{d\times 1}$ $\mathbb{R}^{d\times d}\ \mathbb{R}^{d\times 1}$ $\mathbb{R}^{1\times d}$ $\mathbb{R}^{1\times d}$

Transformer-xl模型计算过程总结

- 一个N层Transformer-XL，有1个attention head的计算过程可以总结如下：

For $n = 1, \ldots, N$:

$$\widetilde{\mathbf{h}}_{\tau}^{n-1} = \left[\mathrm{SG}(\mathbf{m}_{\tau}^{n-1}) \circ \mathbf{h}_{\tau}^{n-1}\right]$$

$\tilde{h}_{\tau+1}^{n-1} \epsilon \mathbb{R}^{(M+L)\times d}$ concat memory

$$\mathbf{q}_{\tau}^{n}, \mathbf{k}_{\tau}^{n}, \mathbf{v}_{\tau}^{n} = \mathbf{h}_{\tau}^{n-1}\mathbf{W}_{q}^{n\top}, \widetilde{\mathbf{h}}_{\tau}^{n-1}\mathbf{W}_{k,E}^{n\top}, \widetilde{\mathbf{h}}_{\tau}^{n-1}\mathbf{W}_{v}^{n\top}$$

attention head

calculate attention

$$\mathbf{A}_{\tau,i,j}^{n} = \mathbf{q}_{\tau,i}^{n\top}\mathbf{k}_{\tau,j}^{n} + \mathbf{q}_{\tau,i}^{n\top}\mathbf{W}_{k,R}^{n}\mathbf{R}_{i-j} + u^{\top}\mathbf{k}_{\tau,j} + v^{\top}\mathbf{W}_{k,R}^{n}\mathbf{R}_{i-j}$$

$$\mathbf{a}_{\tau}^{n} = \text{Masked-Softmax}(\mathbf{A}_{\tau}^{n})\mathbf{v}_{\tau}^{n}$$

$q \in \mathbb{R}^{L\times d}, k, v, R \in \mathbb{R}^{(M+L)\times d}$

$$\mathbf{o}_{\tau}^{n} = \text{LayerNorm}(\text{Linear}(\mathbf{a}_{\tau}^{n}) + \mathbf{h}_{\tau}^{n-1})$$

$$\mathbf{h}_{\tau}^{n} = \text{Positionwise-Feed-Forward}(\mathbf{o}_{\tau}^{n})$$

附录：对有相对位置编码的注意力的高效运算

$\mathbb{R}^{d\times d}$ $\mathbb{R}^{d\times 1}$

- 对每一对(i,j)计算$W_{k,R}R_{i-j}$是一种平方成本，不用每一对(i,j) 都计算。把所有可能的$i-j$计算好，对于不同的$query_i$平移就好了。

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}^{\top}_{x_i}\mathbf{W}^{\top}_{q}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)} + \underbrace{u^{\top}\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(c)} + \underbrace{v^{\top}\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(d)}.$$

- 相对距离$i-j$，只能是0到$M+L-1$之间的整数，M是memory的长度，L是segment的长度。

附录：对有相对位置编码的注意力的高效运算

定义Q如下，注意到Q已经包含了$W_{k,R}R_{i-j}$中每一对(i, j)可能产生的输出向量了

$$\mathbf{Q} := \begin{bmatrix} \mathbf{R}_{M+L-1}^\top \\ \mathbf{R}_{M+L-2}^\top \\ \vdots \\ \mathbf{R}_1^\top \\ \mathbf{R}_0^\top \end{bmatrix} \mathbf{W}_{k,R}{}^\top = \begin{bmatrix} [\mathbf{W}_{k,R}\mathbf{R}_{M+L-1}]^\top \\ [\mathbf{W}_{k,R}\mathbf{R}_{M+L-2}]^\top \\ \vdots \\ [\mathbf{W}_{k,R}\mathbf{R}_1]^\top \\ [\mathbf{W}_{k,R}\mathbf{R}_0]^\top \end{bmatrix} \in \mathbb{R}^{(M+L)\times d}$$

consist of all possible vector outputs of $\mathbf{W}_{k,R}\mathbf{R}_{i-j}$ for any (i, j). Note that we have defined $\mathbf{Q}$ in a reversed order, i.e., $\mathbf{Q}_k = \mathbf{W}_{k,R}\mathbf{R}_{M+L-1-k}$, to make further discussion easier.

附录：对有相对位置编码的注意力的高效运算

将注意力算式的b项展开，展开为$L\times(M+L)$注意力矩阵，如下：

Query: L
Key: M+L

Next, we collect the term (b) for all possible i, j into the following $L \times (M + L)$ matrix,

$$\mathbf{B} = \begin{bmatrix} q_0^\top \mathbf{W}_{k,R}\mathbf{R}_M & \cdots & q_0^\top \mathbf{W}_{k,R}\mathbf{R}_0 & 0 & \cdots & 0 \\ q_1^\top \mathbf{W}_{k,R}\mathbf{R}_{M+1} & \cdots & q_1^\top \mathbf{W}_{k,R}\mathbf{R}_1 & q_1^\top \mathbf{W}_{k,R}\mathbf{R}_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ q_{L-1}^\top \mathbf{W}_{k,R}\mathbf{R}_{M+L-1} & \cdots & q_{L-1}^\top \mathbf{W}_{k,R}\mathbf{R}_{M+L-1} & q_{L-1}^\top \mathbf{W}_{k,R}\mathbf{R}_{L-1} & \cdots & q_{L-1}^\top \mathbf{W}_{k,R}\mathbf{R}_0 \end{bmatrix}$$

$$= \begin{bmatrix} q_0^\top \mathbf{Q}_{L-1} & \cdots & q_0^\top \mathbf{Q}_{M+L-1} & 0 & \cdots & 0 \\ q_1^\top \mathbf{Q}_{L-2} & \cdots & q_1^\top \mathbf{Q}_{M+L-2} & q_1^\top \mathbf{Q}_{M+L-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ q_{L-1}^\top \mathbf{Q}_0 & \cdots & q_{L-1}^\top \mathbf{Q}_M & q_{L-1}^\top \mathbf{Q}_{M+1} & \cdots & q_{L-1}^\top \mathbf{Q}_{M+L-1} \end{bmatrix}$$

附录：对有相对位置编码的注意力的高效运算

- 定义$\tilde{B} = qQ^{\intercal}$。将$\tilde{B}$的每一行分别向左移动即可得到$B$。

$$\widetilde{\mathbf{B}} = \mathbf{q}\mathbf{Q}^{\top} = \begin{bmatrix} q_0^{\top}\mathbf{Q}_0 & \cdots & q_0^{\top}\mathbf{Q}_M & q_0^{\top}\mathbf{Q}_{M+1} & \cdots & q_0^{\top}\mathbf{Q}_{M+L-1} \\ q_1^{\top}\mathbf{Q}_0 & \cdots & q_1^{\top}\mathbf{Q}_M & q_1^{\top}\mathbf{Q}_{M+1} & \cdots & q_1^{\top}\mathbf{Q}_{M+L-1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ q_{L-1}^{\top}\mathbf{Q}_0 & \cdots & q_{L-1}^{\top}\mathbf{Q}_M & q_{L-1}^{\top}\mathbf{Q}_{M+1} & \cdots & q_{L-1}^{\top}\mathbf{Q}_{M+L-1} \end{bmatrix}.$$

- 用同样的方式简化d项的运算。
- 主要的计算成本来自于$qQ^{\intercal}$，正比于$L\times(M+L)\times d$
- 注：按照定义进行矩阵乘法的时间复杂度是$O(abc)$

The End !

https://zhuanlan.zhihu.com/p/92017824
https://www.zhihu.com/question/347678607
https://www.zhihu.com/question/350116316