Transformer-XL

相对位置编码

在 Transformer 中，一个重要的地方在于其考虑了序列的位置信息。在分段的情况下，如果仅仅对于每个段直接使用 Transformer 中的位置编码，即每个不同段在同一个位置上的表示使用相同的位置编码，就会出现问题。例如，第 $i-2$ 段和第 $i-1$ 段的第一个位置将具有相同的位置编码，但它们对于第 i 段的建模重要性显然并不相同。因此，需要对这种位置进行区分。

Transformer-XL 对于这个问题，提出了一种新的位置编码方式，即根据词之间的相对距离而非像 Transfromer 中的绝对位置进行编码。

在 Transformer 中，self-attention 可以表示为：

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

考虑到词嵌入，Q^TK 的完整表达式为：

$$\mathbf{A}_{i,j}^{abs}=(\mathbf{W}_q(\mathbf{E}_{x_i}+\mathbf{U}_i))^T(\mathbf{W}_k(\mathbf{E}_{x_j}+\mathbf{U}_j))$$

将其展开可得：

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

Transformer-XL 在上式基础上做了若干变化，得到下面的计算方法：

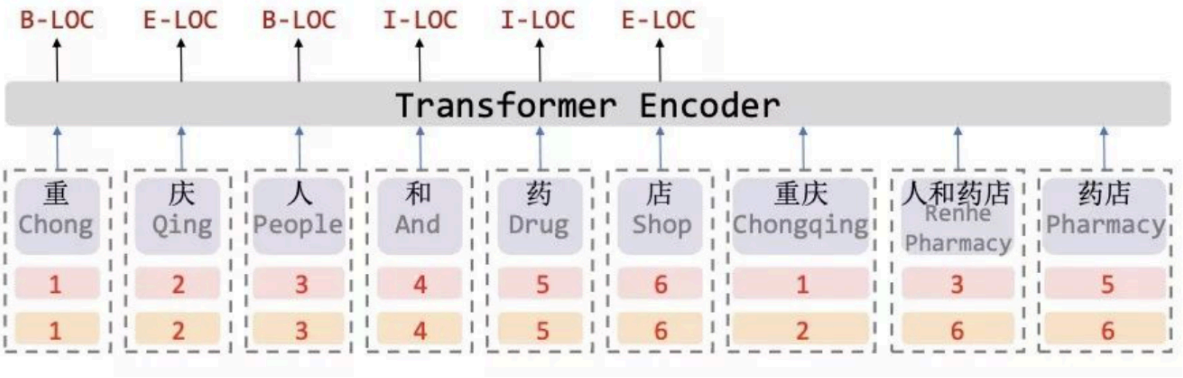
$$\mathbf{A}_{i,j}^{rel}=\underbrace{\mathbf{E}_{x_i}^T\mathbf{W}_q^T\mathbf{W}_{k,E}\mathbf{E}_{x_j}}_{(a)}+\underbrace{\mathbf{E}_{x_i}^T\mathbf{W}_q^T\mathbf{W}_{k,R}\mathbf{R}_{i-j}}_{(b)}+\underbrace{u^T\mathbf{W}_{k,E}^T\mathbf{E}_{x_j}}_{(c)}+\underbrace{v^T\mathbf{W}_{k,R}^T\mathbf{R}_{i-j}}_{(d)}$$

- 第一个变化出现在了 (a)，(b)，(c)，(d) 中： W_k 被拆分成立 $W_{k,E}$ 和 $W_{k,R}$，也就是说输入序列和位置编码不再共享权值。
- 第二个变化是 (b) 和 (d) 中将绝对位置编码 U_j 换成了相对位置编码 R_{i-j} ， 与Transformer一样，这是一个固定的编码向量，不需要学习。
- 第三个变化是 (c)，(d) 中引入了两个新的可学习的参数 $u\in\mathbb{R}^d$ 和 $v\in\mathbb{R}^d$ 来替换Transformer 中的 query 向量 $\mathbf{U}_i^T\mathbf{W}_q^T$。表明**对于所有的 query 位置对应的 query (位置)向量是相同的。** 即无论 query 位置如何，对不同词的注意偏差都保持一致。

改进之后的四个部分也有了各自的含义：

- (a) 没有考虑位置编码的原始分数，只是基于内容的寻址；
- (b) 相对于当前内容的位置偏差；
- (c) 从内容层面衡量键的重要性，表示全局的内容偏置；
- (d) 从相对位置层面衡量键的重要性，表示全局的位置偏置。

FLAT



如上图所示，FLAT 将句子中潜在的词汇拼接到句子的后面，这样在做 NER 任务的时候这些词汇能够给模型提供丰富的信息。引入词汇的同时需要给每个词汇相应的位置信息，这里 FLAT 采用 $head[i]$ 表示词/字（后文统一表示为 span）的起始位置，用 $tail[i]$ 来表示 span 的结束位置。

FLAT 使用了两个位置编码（head position encoding 和 tail position encoding），那么是否可以采用绝对位置编码呢？同样来自邱锡鹏老师组的论文 *TENER: Adapting Transformer Encoder for Named Entity Recognition* 给出答案：**原生Transformer中的绝对位置编码并不直接适用于NER任务**。

TENER 论文发现：**对于NER任务来说，位置和方向信息是十分重要的。** 在「Inc.」前的单词更可能的实体类型是「ORG」，在「in」后的单词更可能为时间或地点。而对于方向性的感知会帮助单词识别其邻居是否构成一个连续的实体 Span。可见，对于「距离」和「方向性」的感知对于 Transformer 适用于 NER 任务至关重要。

但是，**原生 Transformer 的绝对位置编码本身缺乏方向性，虽然具备距离感知，但还是被 self-attention 机制打破了**。

仔细分析，BiLSTM 在 NER 任务上的成功，一个关键就是 BiLSTM 能够区分其上下文信息的方向性，来自左边还是右边。而对于 Transformer，其区分上下文信息的方向性是困难的。因此，要想解决 Transformer 对于 NER 任务表现不佳的问题，必须提升 Transformer 的位置感知和方向感知。

因此，FLAT 采用 Transformer-XL 中提出的相对位置编码来计算 attention score:

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

论文中提出四种相对距离表示 x_i 和 x_j 之间的关系，同时也包含字符和词汇之间的关系：

$$\begin{aligned} d_{ij}^{(hh)} &= head[i] - head[j] \\ d_{ij}^{(ht)} &= head[i] - tail[j] \\ d_{ij}^{(th)} &= tail[i] - head[j] \\ d_{ij}^{(tt)} &= tail[i] - tail[j] \end{aligned}$$

$d_{ij}^{(hh)}$ 表示 x_i 的 head 到 x_j 的 head 的距离，其余类似。相对位置 encoding 表达式为：

$$R_{ij} = RELU(W_r(P_{d_{ij}^{(hh)}} \oplus P_{d_{ij}^{(ht)}} \oplus P_{d_{ij}^{(th)}} \oplus P_{d_{ij}^{(tt)}}))$$

p_d 的计算方式与 Transformer 相同：

$$\mathbf{P}_d^{(2k)} = \sin(d/10000^{2k/d_{model}})$$
$$\mathbf{P}_d^{(2k+1)} = \cos(d/10000^{2k/d_{model}})$$

这里 d 表示字符在句子中的位置，k 表示字符对应的 embedding 的分量。

Sinusoidal Position Encoding

使用正余弦函数表示绝对位置，通过两者乘积得到相对位置：

$$PE(pos, 2i) = \sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}})$$
$$PE(pos, 2i+1) = \cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}})$$

这样设计的好处是**位置 $pos+k$ 的 positional encoding 可以被位置 pos 线性表示，反映其相对位置关系**。

这里需要用到正弦余弦公式：

$$\sin(\alpha+\beta) = \sin\alpha\cdot\cos\beta + \cos\alpha\cdot\sin\beta$$
$$\cos(\alpha+\beta) = \cos\alpha\cdot\cos\beta - \sin\alpha\cdot\sin\beta$$

对于位置 $pos+k$ 的 positional encoding

$$PE_{(pos+k,2i)} = \sin(w_i\cdot(pos+k)) = \sin(w_i pos)\cos(w_i k) + \cos(w_i pos)\sin(w_i k)$$
$$PE_{(pos+k,2i+1)} = \cos(w_i\cdot(pos+k)) = \cos(w_i pos)\cos(w_i k) - \sin(w_i pos)\sin(w_i k)$$

其中，$w_i = \frac{1}{10000^{\frac{2i}{d_{model}}}}$

对上式稍作调整就有：

$$PE_{(pos+k,2i)} = \cos(w_i k)PE_{(pos,2i)} + \sin(w_i k)PE_{(pos,2i+1)}$$
$$PE_{(pos+k,2i+1)} = \cos(w_i k)PE_{(pos,2i+1)} - \sin(w_i k)PE_{(pos,2i)}$$

注意，pos 和 $pos+k$ 相对距离 k 是常数，所以有

$$\begin{bmatrix} PE_{(pos+k,2i)} \\ PE_{(pos+k,2i+1)} \end{bmatrix} = \begin{bmatrix} u & v \\ -v & u \end{bmatrix} \times \begin{bmatrix} PE_{(pos,2i)} \\ PE_{(pos,2i+1)} \end{bmatrix}$$

其中，$u = \cos(w_i\cdot k), v = \sin(w_i\cdot k)$ 为常数，所以 PE_{pos+k} 可以被 PE_{pos} 线性表示

计算 PE_{pos+k} 和 PE_{pos} 的内积，有

$$\begin{aligned} PE_{pos}\cdot PE_{pos+k} &= \sum_{i=0}^{\frac{d}{2}-1} \sin(w_i pos)\cdot\sin(w_i(pos+k)) + \cos(w_i pos)\cdot cos(w_i(pos+k)) \\ &= \sum_{i=0}^{\frac{d}{2}-1} \cos(w_i(pos-(pos+k))) \\ &= \sum_{i=0}^{\frac{d}{2}-1} \cos(w_i k) \end{aligned}$$

其中，$w_i = \frac{1}{10000^{\frac{2i}{d_{model}}}}$

PE_{pos+k} 和 PE_{pos} 的内积会随着相对位置的递增而减小，从而表征位置的相对距离。但是不难发现，由于距离的对称性，Sinusoidal Position Encoding 虽然能够反映相对位置的距离关系，但是无法区分方向。

$$PE_{pos+k}PE_{pos} = PE_{pos-k}PE_{pos}$$

参考资料

1. https://blog.csdn.net/magical_bubble/article/details/89060213
2. https://zhuanlan.zhihu.com/p/271984518
3. https://zhuanlan.zhihu.com/p/225238667
4. https://zhuanlan.zhihu.com/p/509248057
5. https://zhuanlan.zhihu.com/p/121126531