NLP-XLNet

2019年7月3日　　8:36

XLNet模型是一种排列语言模型。

给定长度为 T 的序列 $\mathbf{x}$，总共有 $T!$ 种排列方法，也就对应 $T!$ 种链式分解方法。比如假设 $\mathbf{x}=x1x2x3$，那么总共用 3!=6 种分解方法：

$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_1x_2) \Rightarrow 1 \rightarrow 2 \rightarrow 3$$
$$p(\mathbf{x}) = p(x_1)p(x_2|x_1x_3)p(x_3|x_1) \Rightarrow 1 \rightarrow 3 \rightarrow 2$$
$$p(\mathbf{x}) = p(x_1|x_2)p(x_2)p(x_3|x_1x_2) \Rightarrow 2 \rightarrow 1 \rightarrow 3$$
$$p(\mathbf{x}) = p(x_1|x_2x_3)p(x_2)p(x_3|x_2) \Rightarrow 2 \rightarrow 3 \rightarrow 1$$
$$p(\mathbf{x}) = p(x_1|x_3)p(x_2|x_1x_3)p(x_3) \Rightarrow 3 \rightarrow 1 \rightarrow 2$$

注意 $p(x2|x1x3)$ 指的是第一个词是 $x1$ 并且第三个词是 $x3$ 的条件下第二个词是 $x2$ 的概率，也就是说原来词的顺序是保持的。如果理解为第一个词是 $x1$ 并且第二个词是 $x3$ 的条件下第三个词是 $x2$，那么就不对了。

排列语言模型会学习各种顺序的猜测方法，如3->1->2，是先猜第三个词，然后根据第三个词猜测第一个词，在根据第一个和第三个词猜测第二个词。这样就解决了顺序只有两种的问题。我们可以遍历这所有可能路径，学习语言模型的参数。新的问题是计算量太大。

e(x): x出现的概率

h(xz<t):是前t个词语的按指定顺序出现的概率

$$p_\theta(X_{z_t} = x|\mathbf{x}_{z<t}) = \frac{exp(e(x)^T h_\theta(\mathbf{x}_{z<t}))}{\sum_{x'} exp(e(x')^T h_\theta(\mathbf{x}_{z<t}))}$$

g: 是前t个词语的按指定顺序出现,并且下一个词的位置是Zt的概率

$$p_\theta(X_{z_t} = x|\mathbf{x}_{z<t}) = \frac{exp(e(x)^T g_\theta(\mathbf{x}_{z<t}, z_t))}{\sum_{x'} exp(e(x')^T g_\theta(\mathbf{x}_{z<t}, z_t))}$$

位置问题解决之后，新的问题又来了，如果t2在t1下出现概率很低，那同样前提条件直接gg，无法使用。

我们可以引入两个Stream，即两个隐状态。

1. 内容隐状态，它就会标准的 Transformer 一样，既编码上下文（context）也编码的内容。

2. 查询隐状态，它只编码上下文和要预测的位置 zt，但是不包含x zt这个词本身。

为解决计算量过大的问题引入部分预测：由于前面的词上下文较少，预测第一个词时无任何上下文，故难以精确。我们把一个排列z分成两个序列，z<=c和z>c，只有z>c会被预测。使用超参数K,表示只有1/k的token会被预测

*超参数是由人手工设置，无法由网络自动调节的参数

$$\frac{|z| - c}{|z|} = \frac{1}{K}$$

到此为止，XLNet 的核心思想已经比较清楚了：还是使用语言模型，但是为了解决双向上下文的问题，引入了排列语言模型。排列语言模型在预测时需要 target 的位置信息，因此通过引入 Two-Stream，Content 流编码到当前时刻的所有内容，而 Query 流只能参考之前的历史以及当前要预测的位置。最后为了解决计算量过大的问题，对于一个句子，我们只预测后面的 1/K 的词。