Attention is all you need

September 24, 2024

为什么选择这篇文献？

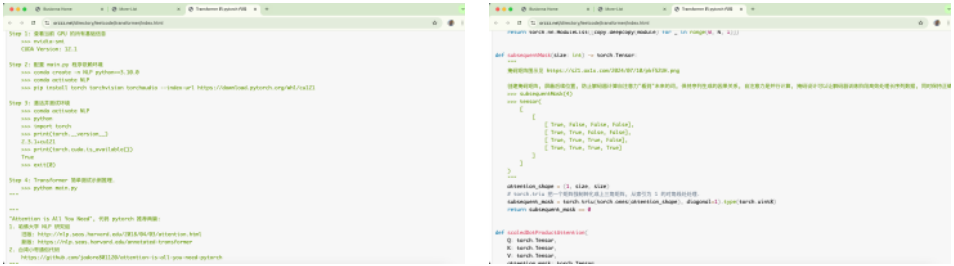

图 1: README.html - 2024-07-10 复现 Transformer 模型所需要的依赖环境 - https://orzzz.net

目录

研究背景

以往的 seq2seq 序列转化

- 基于 Encoder-Decoder 的 RNN 和 CNN 网络
- 时间 $t+1$ 的计算依赖于 t 时刻的结果 $\Longrightarrow$ 限制并行计算能力
- LSTM 算法的局限性 $\Longrightarrow$ 时序过早的信息容易被遗弃
- 整套 RNN 和 CNN 网络时空复杂度高，计算和内存开销大

现在的 Transformer 模型

- 通过 Attention 机制连接 Encoder 和 Decoder，摒弃 RNN 和 CNN 架构

模型架构

Embedding

one-hot vector $\xrightarrow{E}$ meaningful vector

Positional Encoding

trigonometric function

Encoder-Decoder $\times 6$

(Masked) Multi-Head Attention + Norm + FFN

Generator

Linear + log_softmax

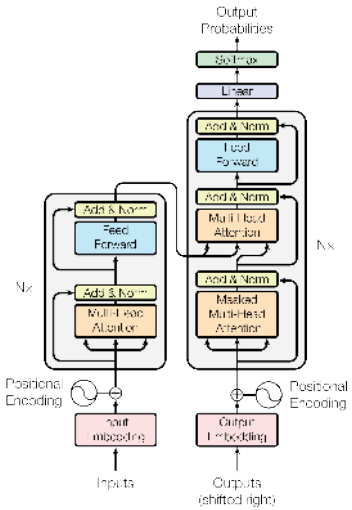

图 2: Transformer 网络

注意力

公式

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

全局信息查询

数据库: 键值对字典 $\{\text{“}Bob\text{”}: 18, \text{“}Cesus\text{”}: 22, \text{“}Bob\text{”}: 20, \text{“}Bylan\text{”}: 19\}$

Q(uery): 查询所有“Bob”的平均年龄 $\Longrightarrow (18 + 20) \div 2 = 19$

Q(uery): 查询首字母为“B”的平均年龄 $\Longrightarrow (18 + 20 + 19) \div 3 = 19$

Q(uery): 查询“&%#@”的平均年龄

注意力

向量相似度化

数据库字典：$\{k_1 : 18, \quad k_2 : 22, \quad k_3 : 20, \quad k_4 : 19\}$

$$\begin{cases} k_1 = [1,\ 2,\ 0] & \#Bob \\ k_2 = [0,\ 0,\ 2] & \#Cesus \\ k_3 = [1,\ 2,\ 0] & \#Bob \\ k_4 = [1,\ 4,\ 0] & \#Bylan \end{cases}$$

注意力查询

约定：$[1,\ 0,\ 0]$ 代表首字母为“B”，Q(uery)：查询首字母为“B”的平均年龄

$$softmax\left\{[1,0,0]\begin{bmatrix}1 & 0 & 1 & 1\\ 2 & 0 & 2 & 4\\ 0 & 2 & 0 & 0\end{bmatrix}\right\}[18,22,20,19]^\top = [\frac{1}{3},0,\frac{1}{3},\frac{1}{3}][18,22,20,19]^\top = 19$$

注意力

多头注意力

卷积层 $\Longrightarrow$ 多个卷积核 $\Longrightarrow$ 多个通道特征
自注意力层 $\Longrightarrow$ 多个注意力 $\Longrightarrow$ 多头自注意力

$$
\begin{aligned}
head_i(E) &= Attention(EQ_{W_i}, EK_{W_i}, EV_{W_i}) \\
MultiHeadSelfAttention(E) &= Contact(head_1, head_2, \ldots, head_h)\mathbb{O}_W \\
MultiHeadSelfAttention(Q, K, V) &= Contact(\\
&\quad Attention(QQ_{W_1}, KK_{W_1}, VV_{W_1}), \\
&\quad Attention(QQ_{W_2}, KK_{W_2}, VV_{W_2}), \\
&\quad \cdots, \\
&\quad Attention(QQ_{W_h}, KK_{W_h}, VV_{W_h}), \\
&\quad)\mathbb{O}_W
\end{aligned}
$$

掩码

掩码矩阵

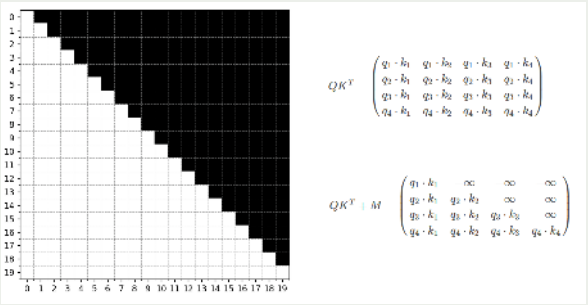

图 3: 黑色部分以 $-\infty$ 代替，softmax 之后注意力权重为 0

掩码

掩码矩阵代码

```python
def subsequentMask(size: int) -> torch.Tensor:
    """
    掩码矩阵图示见 https://s21.ax1x.com/2024/07/10/pkf52UH.png

    创建掩码矩阵，屏蔽后续位置，防止解码器计算自注意力"看到"未来的词，保持序列生成的因果关系，自注意力是并行计算
    >>> subsequentMask(4)
    >>> tensor(
        [
            [
                [ True, False, False, False],
                [ True, True, False, False],
                [ True, True, True, False],
                [ True, True, True, True]
            ]
        ]
    )
    """
    attention_shape = (1, size, size)
    # torch.triu 把一个矩阵强制转化成上三角矩阵，从索引为 1 的对角线处处理.
    subsequent_mask = torch.triu(torch.ones(attention_shape), diagonal=1).type(torch.uint8)
    return subsequent_mask == 0
```

图 4: 输出 $t+1$ 个单词，阻止模型知道 $t+1$ 之后的信息，只保留 t 之前的信息

掩码

推理过程

decoder input: <SOS> decoder output: y_1
decoder input: <SOS> y_1 decoder output: y_2
decoder input: <SOS> y_1 y_2 decoder output: y_3
decoder input: <SOS> y_1 y_2 y_3 decoder output: y_4

掩码过程

<SOS> ~~y_1 y_2 y_3 y_4~~ $\Longrightarrow$ y_1
<SOS> y_1 ~~y_2 y_3 y_4~~ $\Longrightarrow$ y_2
<SOS> y_1 y_2 ~~y_3 y_4~~ $\Longrightarrow$ y_3
<SOS> y_1 y_2 y_3 ~~y_4~~ $\Longrightarrow$ y_4

位置编码

绝对编码

把一个语句中所有单词放到一个集合里，然后从集合的第一个元素直到最后一个元素，标记为 $1, 2, \ldots, n$，但极差会相当大，且无泛化能力

相对编码

在区间 $[0,\ 1]$ 中，给每个单词分配一个浮点数，但不同长度语句中单词差异不一致

理想编码

- 在长度不同的语句中，任意两个单词之间的距离应保持一致

$$f(pos+1) - f(pos) = f(pos) - f(pos-1)$$

- 模型能泛化到更长的句子上，且数值要有限制
- 位置编码必须是确定性的，相同的单词，产生相同的编码

位置编码

二进制编码

从 0 到 15 共 4 列，每列以不同的周期依次出现 0-1

0：	0	0	0	0	8：	1	0	0	0
1：	0	0	0	1	9：	1	0	0	1
2：	0	0	1	0	10：	1	0	1	0
3：	0	0	1	1	11：	1	0	1	1
4：	0	1	0	0	12：	1	1	0	0
5：	0	1	0	1	13：	1	1	0	1
6：	0	1	1	0	14：	1	1	1	0
7：	0	1	1	1	15：	1	1	1	1

图 5: $T_{c1} = 8,\ T_{c2} = 4,\ T_{c3} = 2,\ T_{c4} = 1$

位置编码

三角编码公式

$$PE(pos, 2i) = \sin(pos \div 10000^{2i \div d_{model}})$$

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

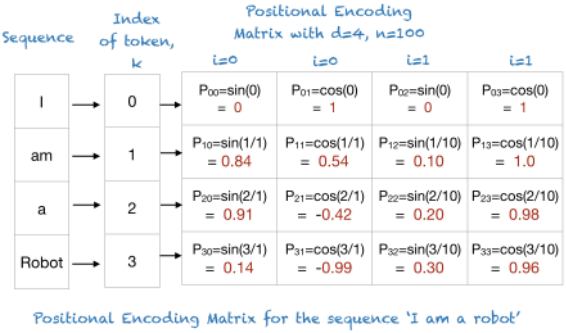

图 6: Positional Encoding

研究结论

Transformer 优点

- 摒弃 RNN 和 CNN 网络，新的架构设计巧妙
- 引用 Attention 运算
- 引入单词位置运算，任意两个单词距离始终是一个单位
- Transformer 模型可并行计算
- 新架构具有启示意义：编码器的 BERT，解码器的 GPT，视觉图像处理的 ViT

Transformer 缺点

- 沿袭传统学习的套路，全连接层 + 注意力层
- 完全摒弃 RNN 和 CNN 后，缺乏捕捉局部特征的能力
- 勉强以 Positional Encoding 弥补词向量位置信息的缺失

Thanks for Listening. ◡̈✓