



人工智能与深度学习

5-3 Transformers

王中雷

经济学院、王亚南经济研究院、邹至庄经济研究院

厦门大学

(第一版)

内容摘要

1. 研究动机
2. 点积自注意力 (Dot-product self-attention)
3. 位置编码
4. 多头自注意力
5. Transformer 层
6. Transformer 模型

研究动机

1. 循环神经网络模型，在序列分析方面取得了成功
 - 循环结构较长的计算路径仍可能增加长期依赖的学习难度
 - 此外，相关模型无法并行，导致其计算效率较低
2. Vaswani et al. (2017) 针对机器翻译任务，提出了多头点积自注意力框架
 - 不再依赖循环神经网络模型结构
 - 该框架适用于并行计算，因此，具有较高的计算效率
 - 更有利于捕捉长距离依赖
 - 该框架被广泛应用于深度学习模型中，用于提取序列信息

Transformers

1. Transformer 是一个新的模型框架
 - 其通过共享模型参数，处理不同长度的输入信息
 - 基于词嵌入，该框架可有效处理不同词向量之间的相关性
2. 其核心是点积自注意力（dot-product self-attention）机制
3. 考虑一句话：“I have arrived in Beijing from Xiamen.”
 - 假设该句话分词化后对应的词向量 $\mathbf{x}_1, \dots, \mathbf{x}_7$
4. 为了符号简单，我们用 $\mathbf{x}_i$ 来代替前面讨论的词向量符号 $\mathbf{x}^{<i>}$

点积自注意力机制

1. 共享模型参数包括: $\{\mathbf{b}_q, \mathbf{W}_q, \mathbf{b}_k, \mathbf{W}_k, \mathbf{b}_v, \mathbf{W}_v\}$

2. 对于每个词向量 $\mathbf{x}_i$, 我们首先计算

- 查询 (Query):

$$\mathbf{q}_i = \mathbf{b}_q + \mathbf{W}_q \cdot \mathbf{x}_i$$

- 键 (Key):

$$\mathbf{k}_i = \mathbf{b}_k + \mathbf{W}_k \cdot \mathbf{x}_i$$

- 值 (Value):

$$\mathbf{v}_i = \mathbf{b}_v + \mathbf{W}_v \cdot \mathbf{x}_i$$

3. 模型参数的维度, 根据如下规则确定:

- 词向量 $\mathbf{x}_i$ 和值 $\mathbf{v}_i$ (一般) 具有相同维度 (在多头自注意力中有所不同)
- 查询 $\mathbf{q}_i$ 和键 $\mathbf{k}_i$ 维度一致, 但可与词向量 $\mathbf{x}_i$ 不同

点积自注意力机制

1. 对于 $i \in \{1, \dots, 7\}$, $\mathbf{x}_i$ 基于如下点积自注意力机制被更新:

- 对于 $l \in \{1, \dots, 7\}$,

$$a_{il} = a(\mathbf{q}_i, \mathbf{k}_l)$$

▷ $a(\mathbf{x}, \mathbf{y})$: 为对齐函数 (alignment function) (Bahdanau et al., 2015), 用于衡量 $\mathbf{x}$ 和 $\mathbf{y}$ 的相关程度

▷ Transformer 用 $a(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \cdot \mathbf{y}$

2. 将 $\{a_{il} : l = 1, \dots, 7\}$ 归一化, 我们得到

$$w_{il} = \frac{\exp(a_{il})}{\sum_{h=1}^7 \exp(a_{ih})}$$

点积自注意力机制

1. 对于 $i \in \{1, \dots, 7\}$, $\mathbf{x}_i$ 基于如下点积自注意力机制被更新（接上一页）：

- 将 $\mathbf{x}_i$ 更新为

$$\mathbf{x}'_i = \sum_{l=1}^7 w_{il} \mathbf{v}_l$$

▷ 在更新 $\mathbf{x}_i$ 时, w_{il} 衡量我们需将多少“注意力”置于 $\mathbf{v}_l$

2. 针对每一个 i , 计算 $\{w_{il} : l = 1, \dots, 7\}$, 并得到更新后的特征向量

$$\{\mathbf{x}'_i : i = 1, \dots, 7\}$$

例子

计算: 更新词向量 $\mathbf{x}_7$

更新: $\mathbf{x}'_1$ $\mathbf{x}'_2$ $\mathbf{x}'_3$ $\mathbf{x}'_4$ $\mathbf{x}'_5$ $\mathbf{x}'_6$ $\mathbf{x}'_7$

归一化:

自注意力:

$\mathbf{q}_1, \mathbf{k}_1, \mathbf{v}_1$ $\mathbf{q}_2, \mathbf{k}_2, \mathbf{v}_2$ $\mathbf{q}_3, \mathbf{k}_3, \mathbf{v}_3$ $\mathbf{q}_4, \mathbf{k}_4, \mathbf{v}_4$ $\mathbf{q}_5, \mathbf{k}_5, \mathbf{v}_5$ $\mathbf{q}_6, \mathbf{k}_6, \mathbf{v}_6$ $\mathbf{q}_7, \mathbf{k}_7, \mathbf{v}_7$

模型参数:

$\mathbf{b}_q, \mathbf{W}_q, \mathbf{b}_k$ $\mathbf{W}_k, \mathbf{b}_v, \mathbf{W}_v$



词嵌入: $\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\mathbf{x}_4$ $\mathbf{x}_5$ $\mathbf{x}_6$ $\mathbf{x}_7$

向量化

1. 记

$$\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_7) \in \mathbb{R}^{d \times 7}$$

$$\mathbf{Q} = \mathbf{b}_q \cdot \mathbf{1}^T + \mathbf{W}_q \cdot \mathbf{X}$$

$$\mathbf{K} = \mathbf{b}_k \cdot \mathbf{1}^T + \mathbf{W}_k \cdot \mathbf{X}$$

$$\mathbf{V} = \mathbf{b}_v \cdot \mathbf{1}^T + \mathbf{W}_v \cdot \mathbf{X}$$

$$\mathbf{X}' = (\mathbf{x}'_1, \dots, \mathbf{x}'_7) \in \mathbb{R}^{d \times 7}$$

2. 点积自注意力机制的向量化为

$$\mathbf{X}' = \mathbf{V} \cdot \text{Softmax}(\mathbf{K}^T \cdot \mathbf{Q})$$

3. 自注意力机制无法考虑句子中，词汇出现的顺序

I arrived in Beijing from Xiamen. I arrived in Xiamen from Beijing.

位置编码

1. Vaswani et al. (2017) 提出，将一个位置编码矩阵 $\mathbf{P}$ ，加到词嵌入后的特征矩阵 $\mathbf{X}$

$$\mathbf{X}_p = \mathbf{X} + \mathbf{P}$$

- $\mathbf{P}$ ：对位置信息进行编码
- $\mathbf{X}_p$ ：加上位置信息后的词嵌入矩阵

2. 我们利用带有不同频率的正余弦函数，得到位置信息矩阵 $\mathbf{P}$

$$\mathbf{P}_{(2i),j} = \sin(j/10000^{2i/d}), \quad \mathbf{P}_{(2i+1),j} = \cos(j/10000^{2i/d})$$

- i ：词向量中的维度指标
- j ：分词所在位置指标

基于放缩的点积自注意力机制

1. 在计算结果 $\mathbf{K}^T \cdot \mathbf{Q}$ 中，可能存在很多绝对值很大的值
2. 使 softmax 进入饱和区域
3. 注意力分布过于尖锐、梯度很小，从而增加优化难度
4. 为了解决这个问题，我们考虑

$$\mathbf{X}' = \mathbf{V} \cdot \text{Softmax} \left(\frac{\mathbf{K}^T \cdot \mathbf{Q}}{\sqrt{d_q}} \right)$$

- d_q ：“查询（query）”以及“键（key）”的维度
- 在统计的角度讲，为了数值稳定，我们除掉的是，对应加和的“标准差”的阶

多头自注意力机制

1. 只基于一组“查询（queries）”、“键（keys）”和“值（values）”的模型过于简单
2. 我们可以考虑 H 个集合，用来提取不同信息
3. 记

$$\mathbf{H}'_h = \mathbf{V}_h \cdot \text{Softmax} \left(\frac{\mathbf{K}_h^T \cdot \mathbf{Q}_h}{\sqrt{d_q}} \right) \quad (h = 1, \dots, H)$$

- $\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h$: 第 h 个集合中的“查询（queries）”、“键（keys）”和“值（values）”
4. 沿特征维拼接各个“头”的输出，并利用线性变换，得到最终特征

$$\mathbf{X}' = \mathbf{W}_o \cdot (\mathbf{H}_1^T, \dots, \mathbf{H}_H^T)^T$$

- $\mathbf{W}_o$: 用来融合不同“头”计算结果的（模型参数）权重项

备注

1. 优势:

- **并行**: 与循环神经网络不同, 该结构可通过并行计算, 提升训练效率
- **长距离相关性**: 自注意机制能够简单地处理, 不同距离的相关信息
- **上下文理解**: 有些分词需要放在整个句子中进行分析才有意义, 该机制能够让语言模型更加准确
- **可解释的权重**: 可作为观察模型关注模式的辅助信息, 但不一定构成可靠的因果解释

备注

1. 缺点:

- **计算成本高昂**: 对于含有 (n) 个词元的长序列, 自注意力机制需要计算两两分词间的相互影响, 这导致计算复杂度和内存需求的阶为 $O(n^2d)$
- **对局部信息关注不足**: 全局自注意力缺少卷积那样显式的局部性归纳偏置
- **过拟合**: 模型参数较多, 当语料库规模较小时, 容易产生过拟合

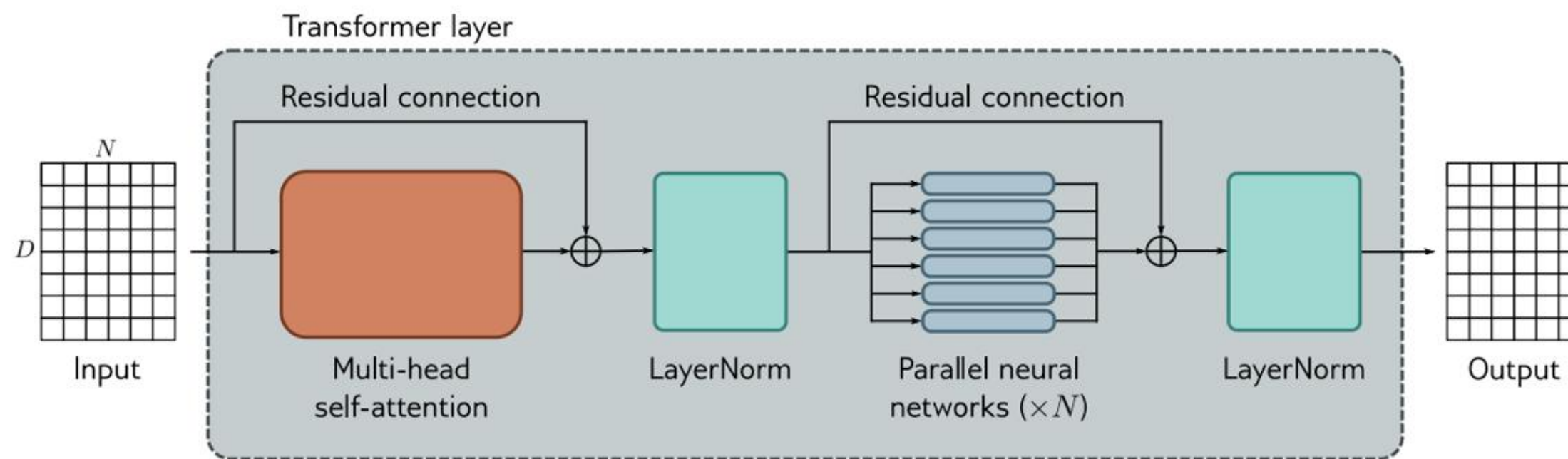
Transformer 层

1. 该层由以下部分组成:

- 多头点积自注意力模块 (提取分词之间的相关性)
- 全连接神经网络 (共享模型参数, 对每个特征向量单独进行)
- 残差连接
- 层归一化

2. 右图来自于 Prince (2024)

中的图 12.7



Transformer 层

1. 计算如下:

$$\mathbf{X} \leftarrow \mathbf{X} + \text{MhSa}(\mathbf{X})$$

$$\mathbf{X} \leftarrow \text{LayerNorm}(\mathbf{X})$$

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \text{MLP}(\mathbf{x}_i)$$

$$\mathbf{X} \leftarrow \text{LayerNorm}(\mathbf{X})$$

- $\mathbf{X}$: 对于单一输入句子, 融合了位置编码信息的词嵌入矩阵
- $\mathbf{x}_i$: 矩阵 $\mathbf{X}$ 的第 i 列 (与之前的设计矩阵不同)
- $\text{MhSa}(\mathbf{X})$: 多头点积自注意力模块
- $\text{MLP}(\mathbf{x})$: 全连接神经网络模块 (共享模型参数)
- $\text{LayerNorm}(\mathbf{X})$: 层归一化模块

Transformer 层

1. 回顾批量归一化

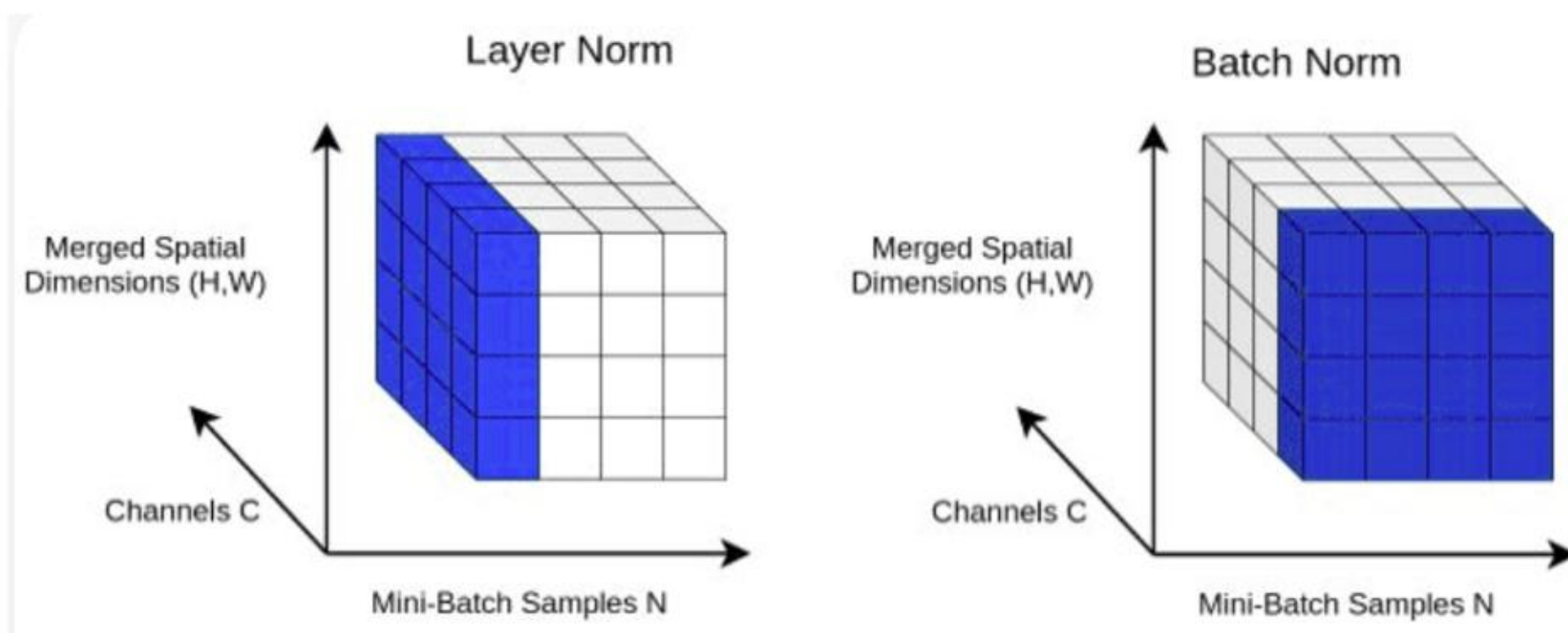
- 用于同一批次中，对每个特征对应的所有训练样本信息进行归一化
- 由于不同句子长度不同，其通常不被用在序列模型中

Transformer 层

1. 对于每一个输入句子，层归一化分别用于每个词向量的维度

2. 下图来自于<https://theaisummer.com/normalization/>

- N : 同一批次样本量
- C : 特征数量
- H, W : 向量化后的词向量的维度



Transformer 模型

1. 可能包括多个 transformer 模块
2. 编码-解码结构用于分析“序列到序列”的任务
 - 编码（“文本到数字”）：将文本进行数字化处理，并基于模型对其进行分析
 - 解码（“数字到文本”）：基于现有分词，预测下一个分词
3. 例子
 - BERT：一个编码模型
 - GPT/ChatGPT：一个采用仅解码器式 Transformer 的语言模型
4. 更多内容请参见 Prince (2024) 的第 12 章