序列到序列模型

李航
字节跳动AI Lab

目录

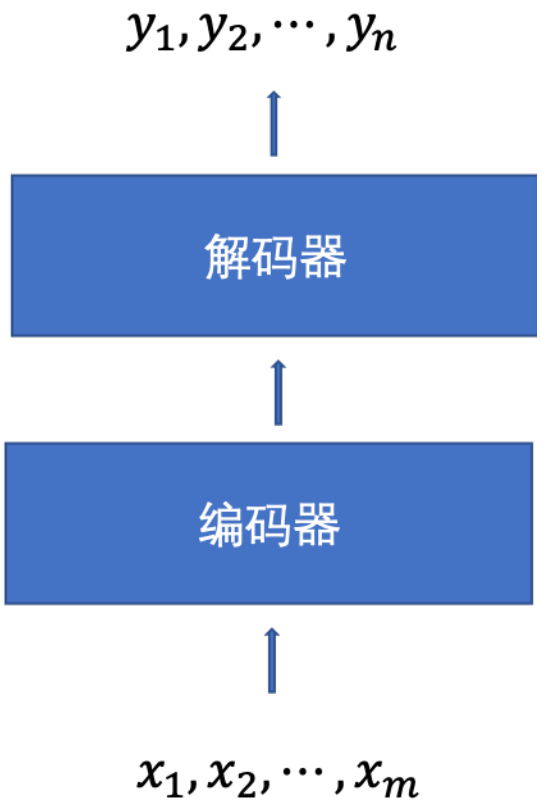
- *序列到序列学习*
- 基本模型
- RNN Search模型
- Transformer模型
- 总结

序列到序列学习

- 模型：编码器-解码器
  - 编码器：输入单词序列到中间表示序列
  - 解码器：中间表示序列到输出单词序列
- 解码：自回归过程，编码：非自回归过程
- 有条件的语言生成
- 机器翻译、摘要生成、对话生成

序列到序列模型

- 编码器-解码器框架



序列到序列模型

- 模型定义
  - $P(y_1, y_2, \cdots, y_n | x_1, x_2, \cdots, x_m) = \prod_{i=1}^{n} P(y_i | y_1, y_2, \cdots, y_{i-1}, x_1, x_2, \cdots, x_m)$
  - $P(y_1, y_2, \cdots, y_n | x_1, x_2, \cdots, x_m) = \prod_{i=1}^{n} P(y_i | \boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{i-1}, \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m)$
- 编码器和解码器
  - $P(y_i | \boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{i-1}, \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m) = P(y_i | \boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{i-1}, \boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_m) P(\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_m | \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m)$

序列到序列学习

- 学习的特点
  - 编码器-解码器联合训练
  - 强制指导
  - 反向传播
- 预测（生成）的特点
  - 束搜索

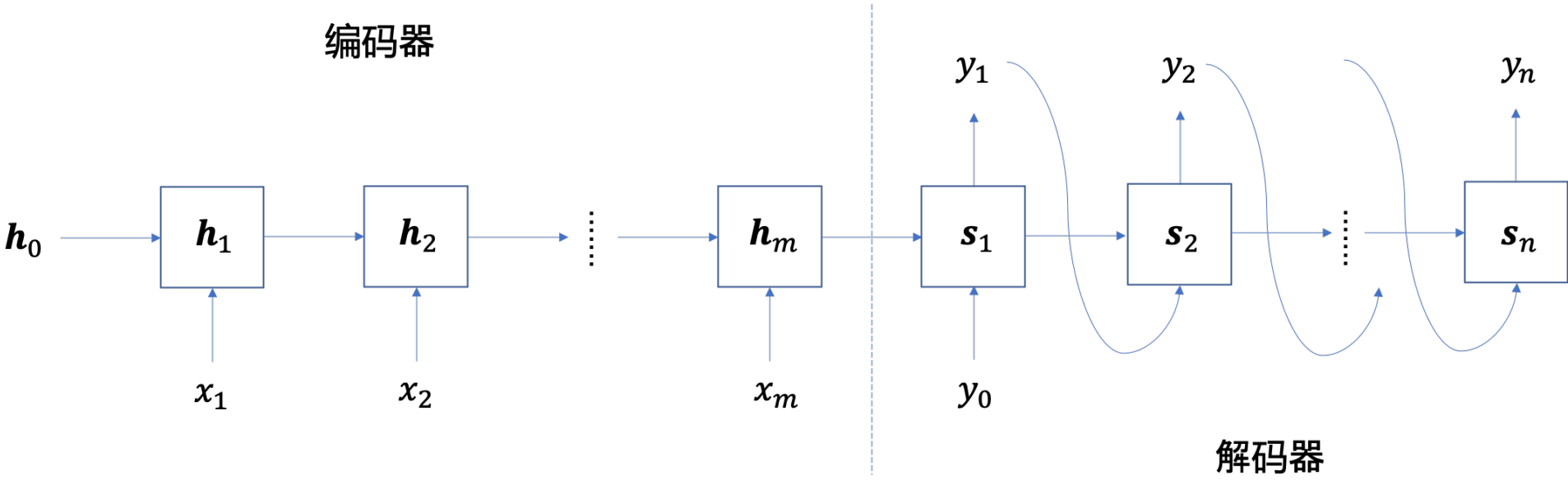
目录

- 序列到序列学习
- *基本模型*
- RNN Search模型
- Transformer模型
- 总结

基本模型

- 模型架构
  - 编码器：LSTM、GRU
  - 解码器：LSTM、GRU
- 中间表示
  - 编码器的最终状态
  - 解码器的初始状态

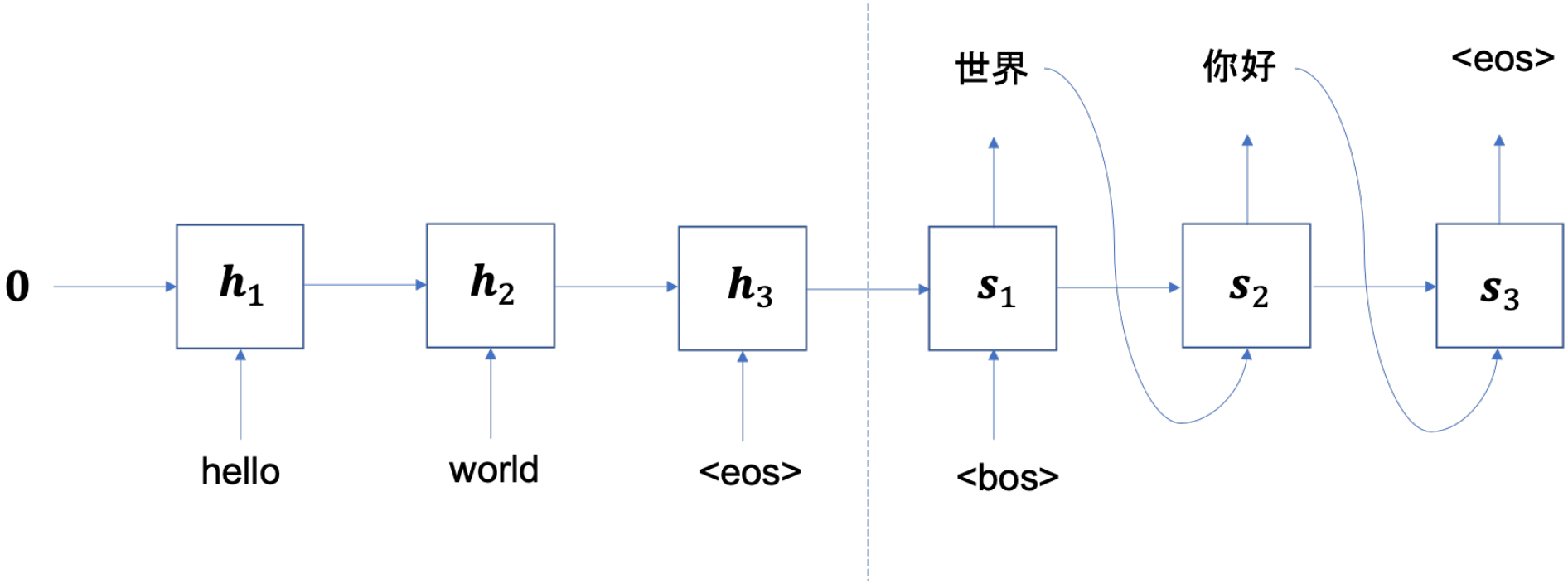
基本模型



基本模型

- 编码器
  - $\boldsymbol{h}_j = a\left(\boldsymbol{x}_j, \boldsymbol{h}_{j-1}\right), \quad j = 1,2,\cdots,m$
- 解码器
  - $\boldsymbol{s}_i = a(\boldsymbol{y}_{i-1}, \boldsymbol{s}_{i-1}), \quad i = 1,2,\cdots,n$
  - $\boldsymbol{p}_i = g(\boldsymbol{s}_i), \quad i = 1,2,\cdots,n$
- 中间表示
  - $\boldsymbol{s}_0 = \boldsymbol{h}_m$

机器翻译



目录

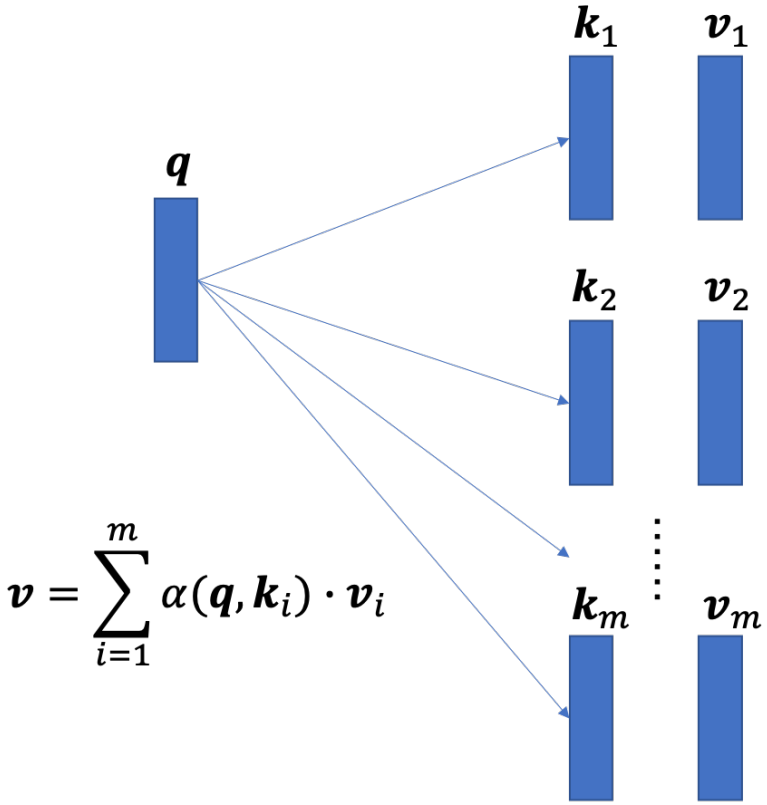
- 序列到序列学习
- 基本模型
- *RNN Search模型*
- Transformer模型
- 总结

RNN Search模型

- 模型架构
  - 编码器：双向LSTM
  - 解码器：单向LSTM
- 中间表示
  - 注意力：动态地、有选择地获取中间表示
- 上下文向量
  - 表示各个位置的注意力计算结果

注意力

- 向量表示的相关性或相似性的计算方法



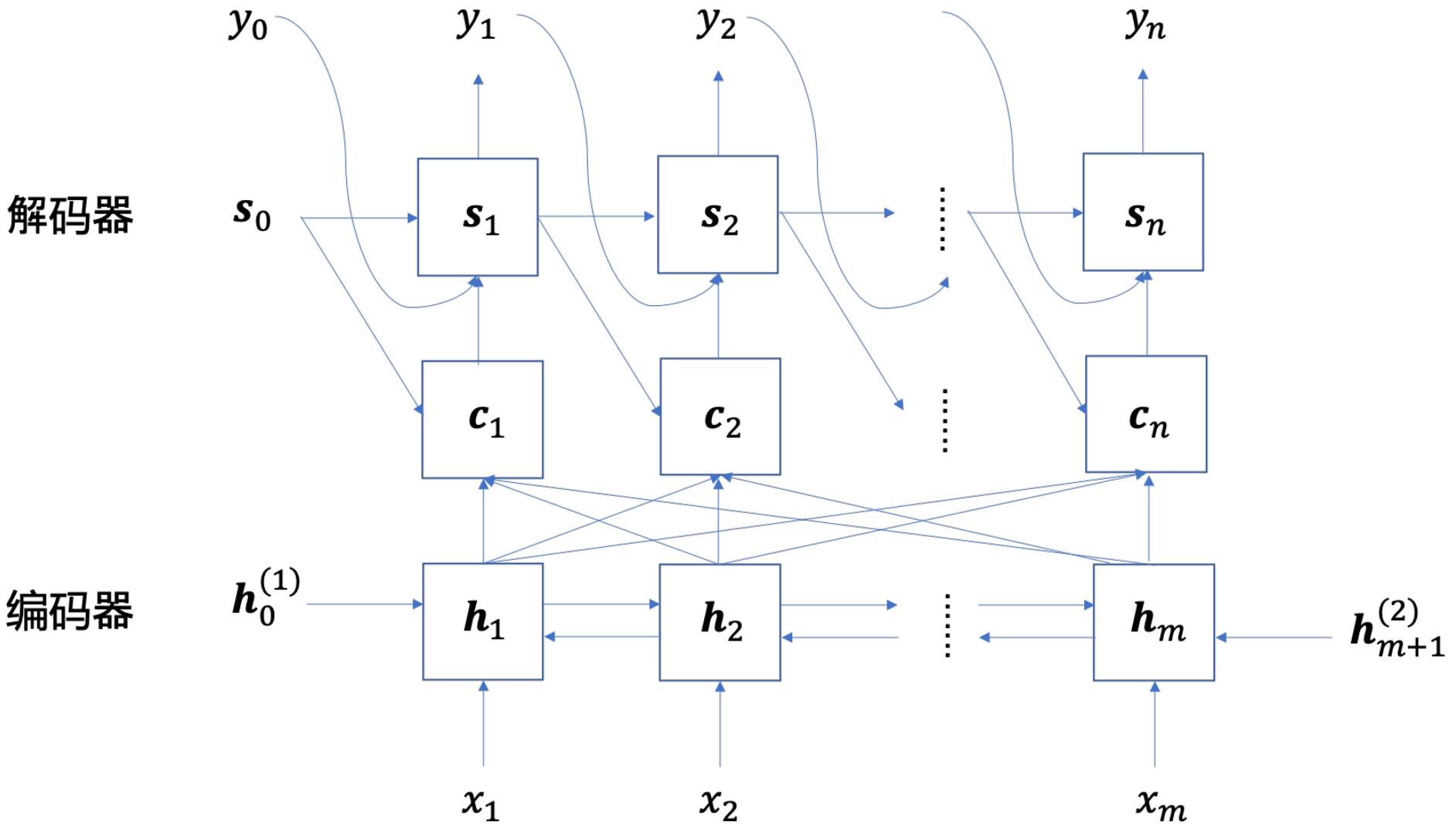
注意力

- 注意力
  - $\boldsymbol{v} = \sum_{i=1}^{n} \alpha(\boldsymbol{q}, \boldsymbol{k}_i) \cdot \boldsymbol{v}_i$
- 注意力权
  - $\alpha(\boldsymbol{q}, \boldsymbol{k}_i) = \text{softmask}(\boldsymbol{q}, \boldsymbol{k}_i) = \frac{e(\boldsymbol{q}, \boldsymbol{k}_i)}{\sum_{j=1}^{n} e(\boldsymbol{q}, \boldsymbol{k}_j)}$
  - $\sum_{i=1}^{n} \alpha(\boldsymbol{q}, \boldsymbol{k}_i) = 1$
- 加法注意力
  - $e(\boldsymbol{q}, \boldsymbol{k}_i) = \sigma\left(\boldsymbol{w}^{\mathrm{T}} \cdot \boldsymbol{q} \oplus \boldsymbol{k}_i + b\right)$
- 乘法注意力
  - $e(\boldsymbol{q}, \boldsymbol{k}_i) = \boldsymbol{q}^{\mathrm{T}} \cdot \boldsymbol{k}_i$
  - $e(\boldsymbol{q}, \boldsymbol{k}_i) = \frac{\boldsymbol{q}^{\mathrm{T}} \cdot \boldsymbol{k}_i}{\sqrt{d}}$

注意力

- 注意力的计算是软的，而不是硬的
- 如果向量都是读热向量，注意力退化为一般的键-值数据库查询
- 注意力模型的复杂度，不随数据规模增大而增大

RNN Search模型



RNN Search模型

- 编码器
  - $\boldsymbol{h}_j^{(1)} = a\left(\boldsymbol{x}_j, \boldsymbol{h}_{j-1}^{(1)}\right), \quad j = 1,2,\cdots,m$
  - $\boldsymbol{h}_j^{(2)} = a\left(\boldsymbol{x}_j, \boldsymbol{h}_{j+1}^{(2)}\right), \quad j = m, m-1,\cdots,1$
  - $\boldsymbol{h}_j = \boldsymbol{h}_j^{(1)} \oplus \boldsymbol{h}_j^{(2)}, \quad j = 1,2,\cdots,m$
- 解码器
  - $\boldsymbol{s}_i = a(\boldsymbol{y}_{i-1}, \boldsymbol{s}_{i-1}, \boldsymbol{c}_i), \quad i = 1,2,\cdots,n$
  - $\boldsymbol{p}_i = g(\boldsymbol{s}_i), \quad i = 1,2,\cdots,n$
- 上下文向量
  - $\boldsymbol{c}_i = \sum_{j=1}^{m} \alpha_{ij}\, \boldsymbol{h}_j, \quad i = 1,2,\cdots,n$
  - $\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{m} \exp(e_{ik})}, \quad i = 1,2,\cdots,n, \;\; j = 1,2,\cdots,m$
  - $e_{ij} = \sigma\left(\boldsymbol{w}^{\mathrm{T}} \cdot \boldsymbol{s}_{i-1} \oplus \boldsymbol{h}_j + b\right), \quad i = 1,2,\cdots,n, \;\; j = 1,2,\cdots,m$

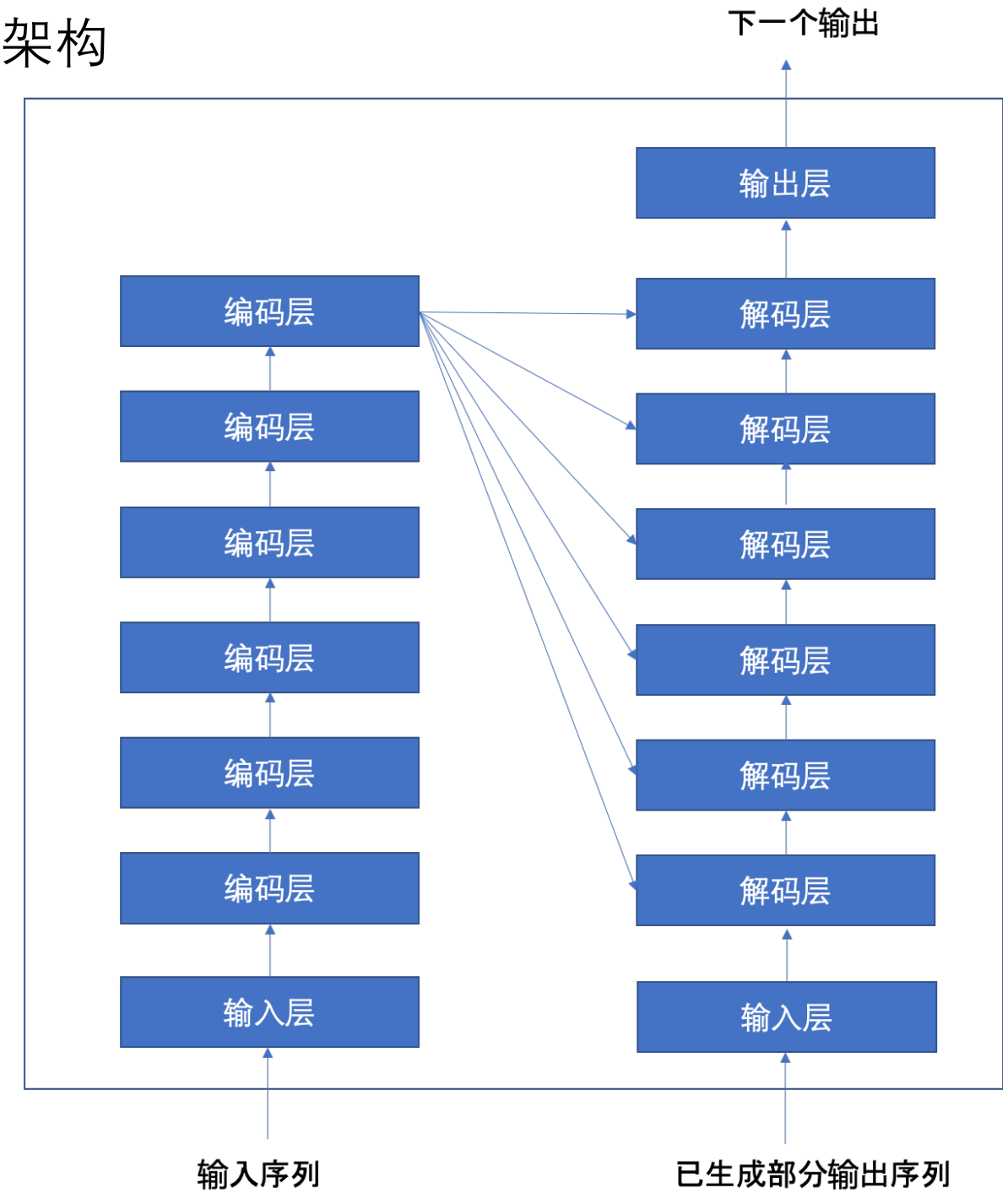
目录

- 序列到序列学习
- 基本模型
- RNN Search模型
- *Transformer模型*
- 总结

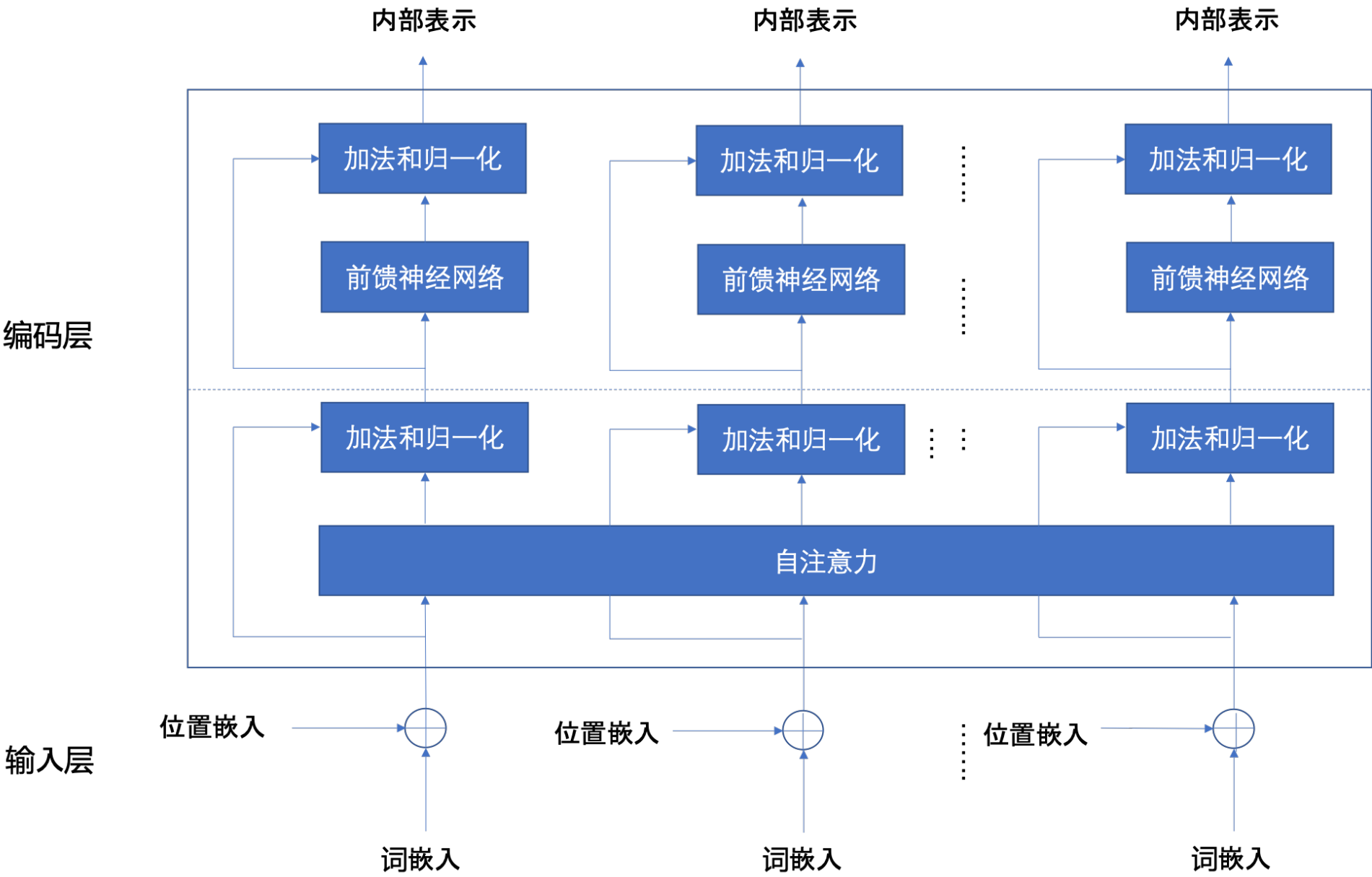
Transformer模型

- Transformer完全使用注意力机制，实现编码器、解码器、以及编码器和解码器之间的信息传递
- Transformer的主要特点
  - 用多头自注意力实现编码器和解码器
  - 用多头注意力实现编码器和解码器之间的信息传递
  - 用掩码（masking）解决解码器的学习问题
  - 用位置编码表示序列的位置信息
  - 用前馈神经网络进行非线性变换
  - 用残差链接和层归一化保证学习的效果。

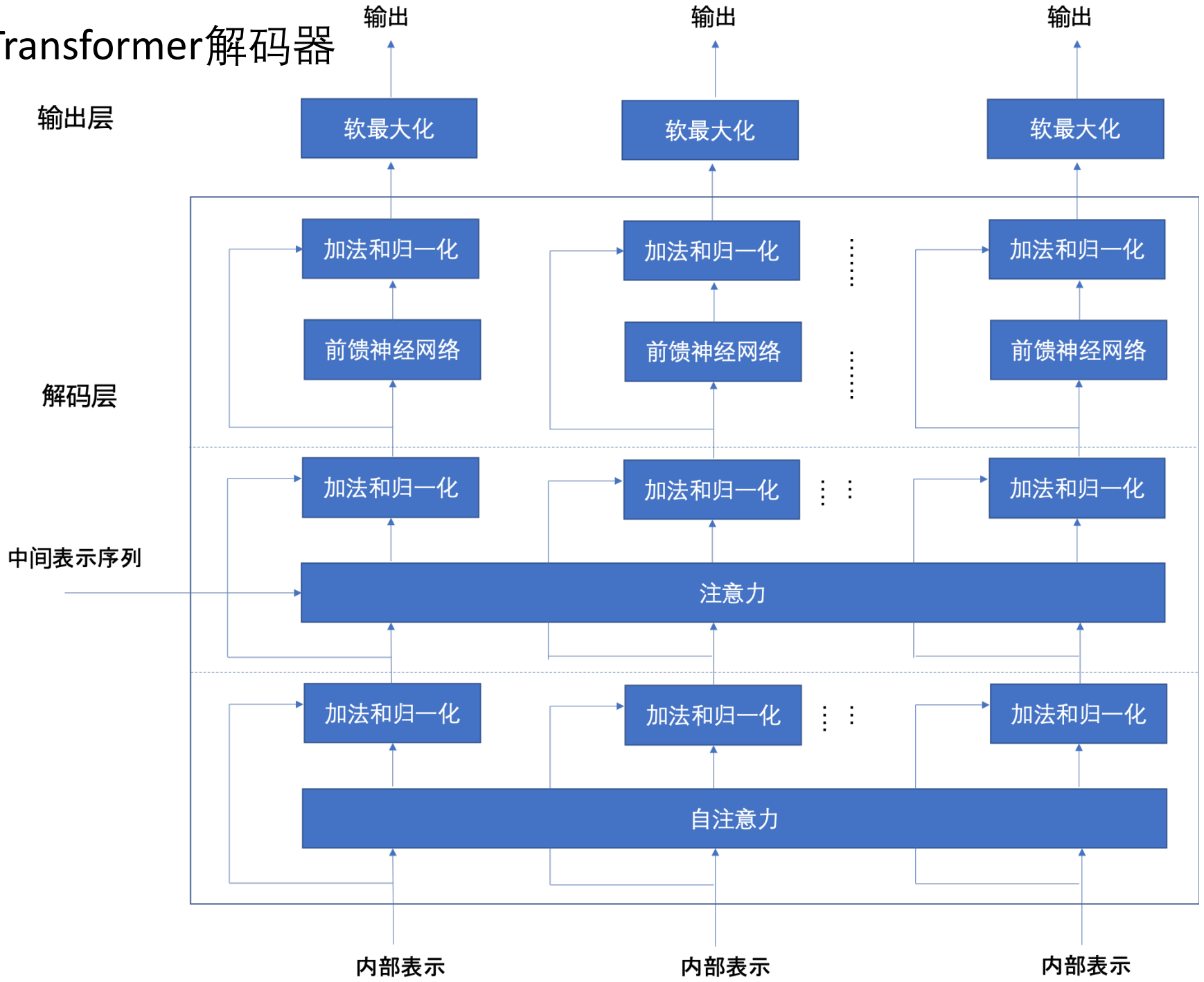
Transformer架构



Transformer编码器

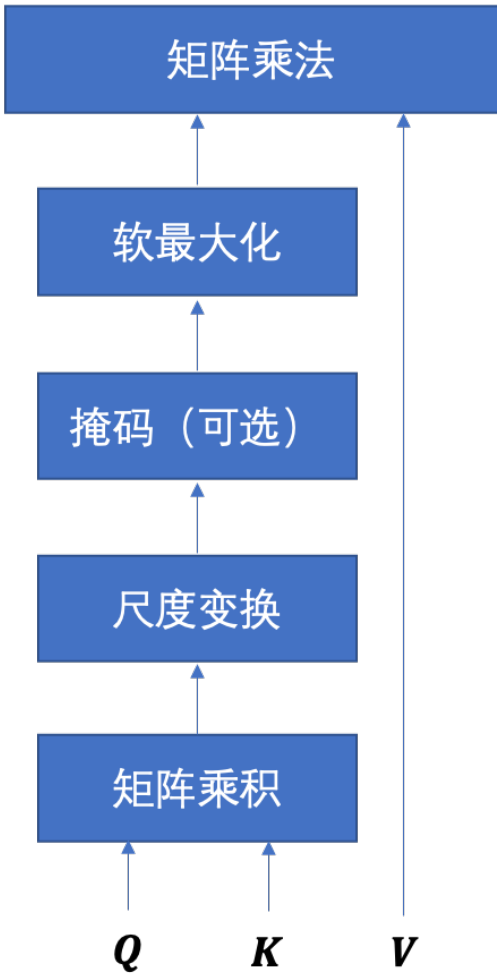


Transformer解码器

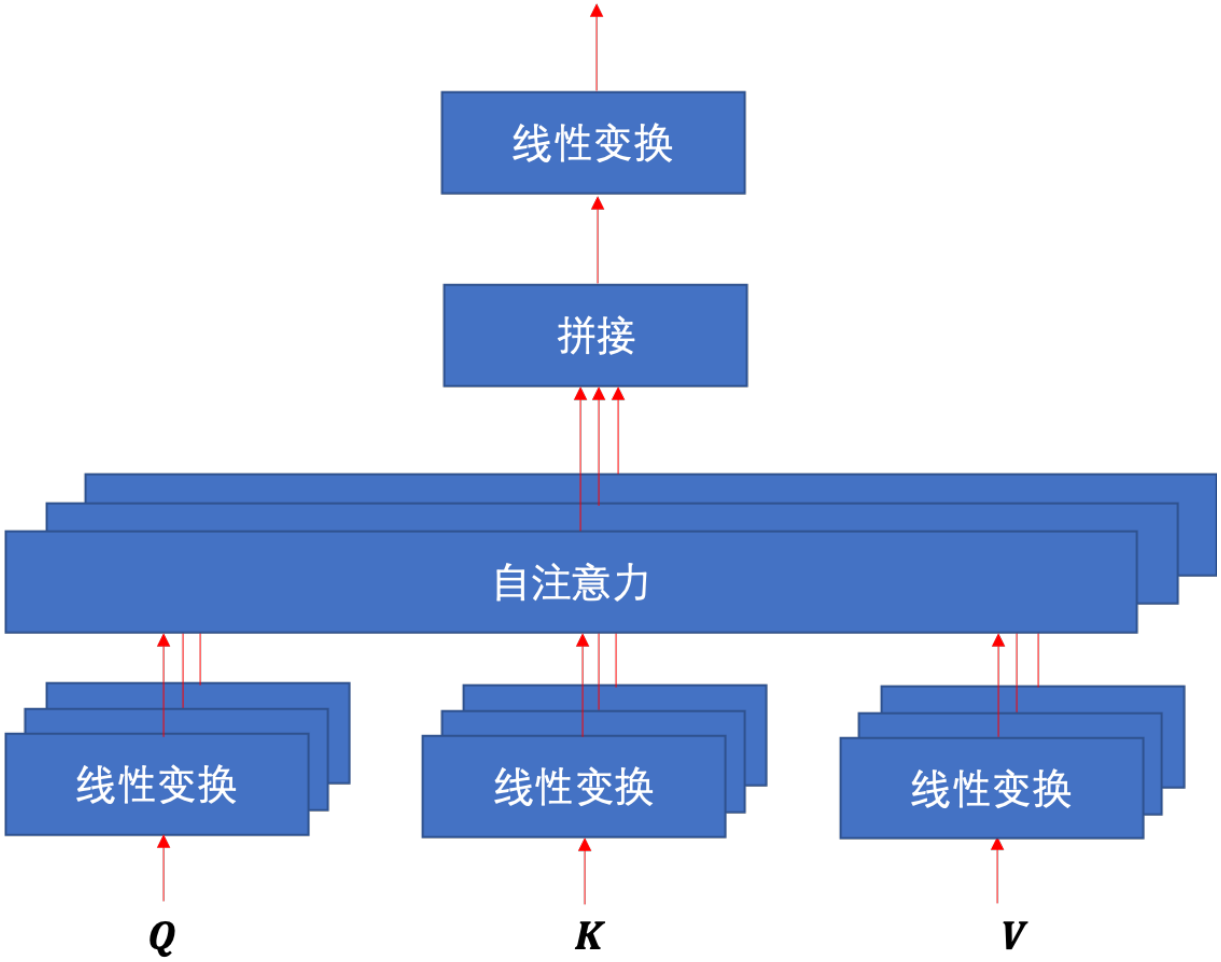


注意力计算

- 多个位置的并行计算
- 矩阵计算



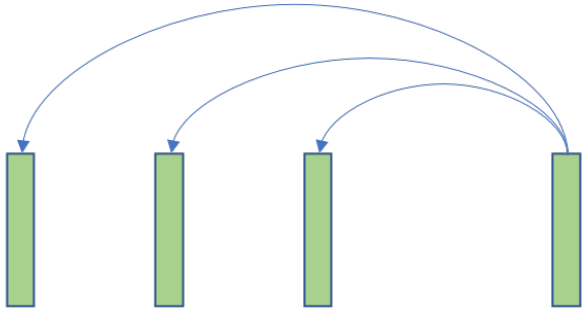
多头注意力计算



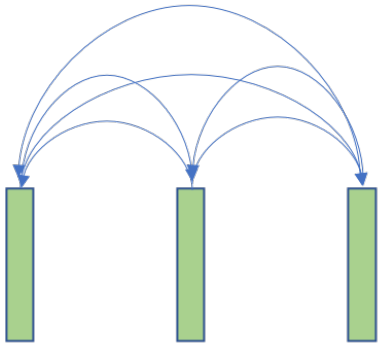
多头注意力

- 注意力
  - $\text{attend}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{V} \cdot \text{softmax}(\frac{\boldsymbol{K}^{\mathrm{T}} \cdot \boldsymbol{Q}}{\sqrt{d}})$
- 多头注意力
  - $\text{multi_head}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{W}_o \cdot \text{concate}(\boldsymbol{U}_1, \boldsymbol{U}_2, \cdots \boldsymbol{U}_h)$
  - $\boldsymbol{U}_i = \text{attend}\left(\boldsymbol{W}_Q^{(i)}\boldsymbol{Q}, \boldsymbol{W}_K^{(i)}\boldsymbol{K}, \boldsymbol{W}_V^{(i)}\boldsymbol{V}\right), \quad i = 1,2, \cdots, h$

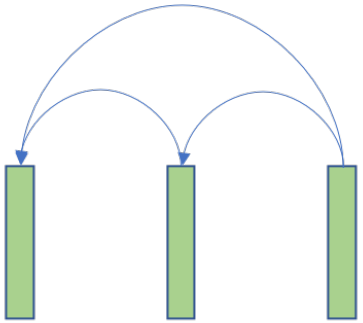
Transformer中的注意力



(a) 注意力



(b) 自注意力



(c) 自注意力

Transformer模型

- 基本函数
  - 输入向量：$\boldsymbol{e}+\boldsymbol{p}$
  - 前馈网络：$\mathrm{forward}(\boldsymbol{z})=\boldsymbol{W}_2\,\mathrm{relu}(\boldsymbol{W}_1\boldsymbol{z}+\boldsymbol{b}_1)+\boldsymbol{b}_2$
  - 残差连接：$\boldsymbol{z}+f(\boldsymbol{z})$
  - 层归一化：$\mathrm{norm}(\boldsymbol{z})=\gamma\frac{\boldsymbol{z}-\boldsymbol{u}}{\sqrt{\sigma^2+\varepsilon}}+\boldsymbol{\beta}$

Transformer模型

- 编码层
  - $\boldsymbol{H}_E^{(0)} = \boldsymbol{E}_E + \boldsymbol{P}_E$
  - $\boldsymbol{Z}_E^{(l)} = \text{norm}(\boldsymbol{H}_E^{(l-1)} + \text{multi_head}(\boldsymbol{H}_E^{(l-1)}, \boldsymbol{H}_E^{(l-1)}, \boldsymbol{H}_E^{(l-1)}))$
  - $\boldsymbol{H}_E^{(l)} = \text{norm}(\boldsymbol{Z}_E^{(l)} + \text{forward}(\boldsymbol{Z}_E^{(l)}))$
- 解码层
  - $\boldsymbol{H}_D^{(0)} = \boldsymbol{E}_D + \boldsymbol{P}_D$
  - $\boldsymbol{I}_D^{(l)} = \text{norm}(\boldsymbol{H}_D^{(l-1)} + \text{multi_head}(\boldsymbol{H}_D^{(l-1)}, \boldsymbol{H}_D^{(l-1)}, \boldsymbol{H}_D^{(l-1)}))$
  - $\boldsymbol{Z}^{(l)} = \text{norm}(\boldsymbol{I}_D^{(l)} + \text{multi_head}(\boldsymbol{I}_D^{(l)}, \boldsymbol{H}_E^{(L)}, \boldsymbol{H}_E^{(L)}))$
  - $\boldsymbol{H}_D^{(l)} = \text{norm}(\boldsymbol{Z}_D^{(l)} + \text{forward}(\boldsymbol{Z}_D^{(l)})$

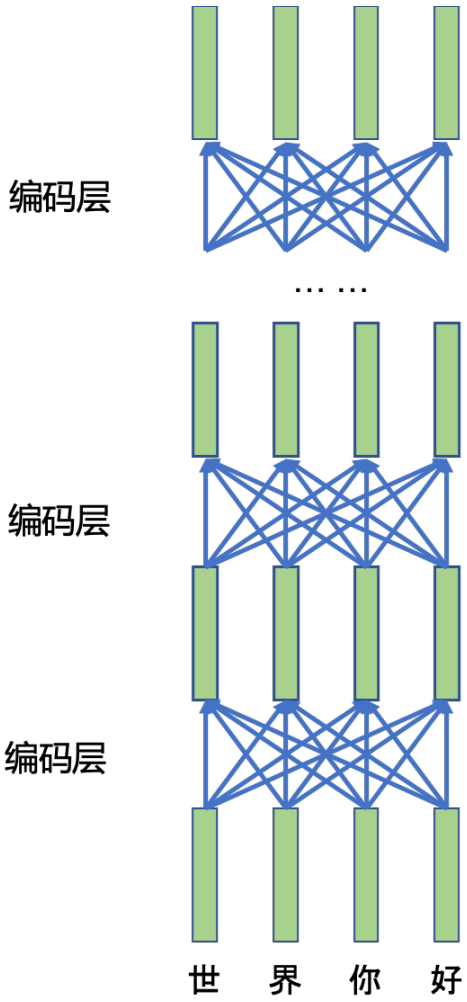
Transformer模型

- 解码层的掩吗
  - $\text{attend}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{V} \cdot \text{softmax}(\frac{\boldsymbol{K}^{\mathrm{T}} \cdot \boldsymbol{Q} + \boldsymbol{M}}{\sqrt{d}})$
  - $\boldsymbol{M} = [m_{ij}],\ \ m_{ij} = \begin{cases} 0 & i \leq j \\ -\infty & \text{otherwise} \end{cases}$

Transformer模型

- 可以刻画自然语言句子的层次化语义结构
- 多头注意力可以表示不同侧面的语义
- 注意力机制能很好地表示单词之间的语义相关性。
- 位置嵌入和残差连接的组合可以表示单词之间的顺序关系
- 可以处理可变长序列。注意力机制不依赖于序列的长度
- 可以自然进行平行处理，计算效率高

Transformer模型的表示学习



目录

- 序列到序列学习
- 基本模型
- RNN Search模型
- Transformer模型
- *总结*

总结

- 序列到序列学习
  - 单词序列到单词序列的转换
  - 编码器-解码器框架
- 基本模型
  - 用RNN实现编码器和解码器
- RNN Search
  - 使用注意力将中间表示从编码器传到解码器
- Transformer
  - 用注意力实现编码器、解码器、编码器到解码器的信息传递
  - 强大的表示能力

We are hiring!

lihang.lh@bytedance.com