self-attention自注意力机制

输入与输出

输入

模型的输入是可变数量的向量集:

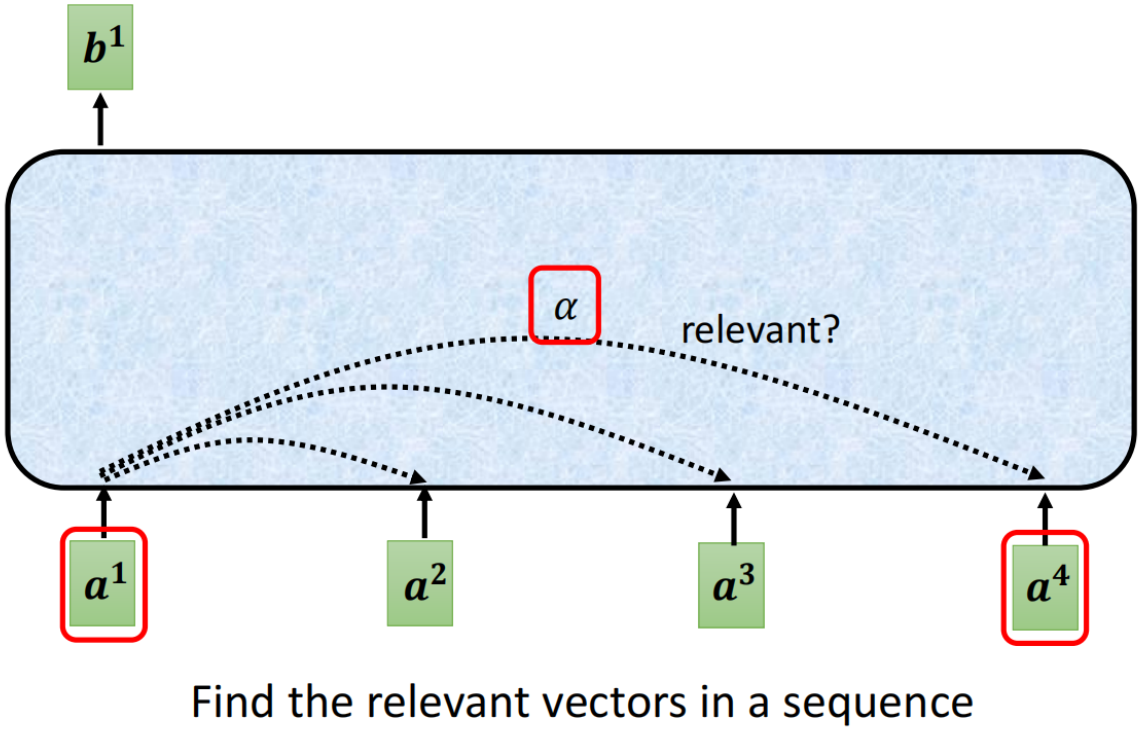
- NLP: 一句话为一个输入, 句中的每个单词为一个向量
- 语音: 按照时间来切割语音, 每一小段为一个向量
- 图网络: 每个node为一个向量
- CV: 每个像素为一个向量

输出

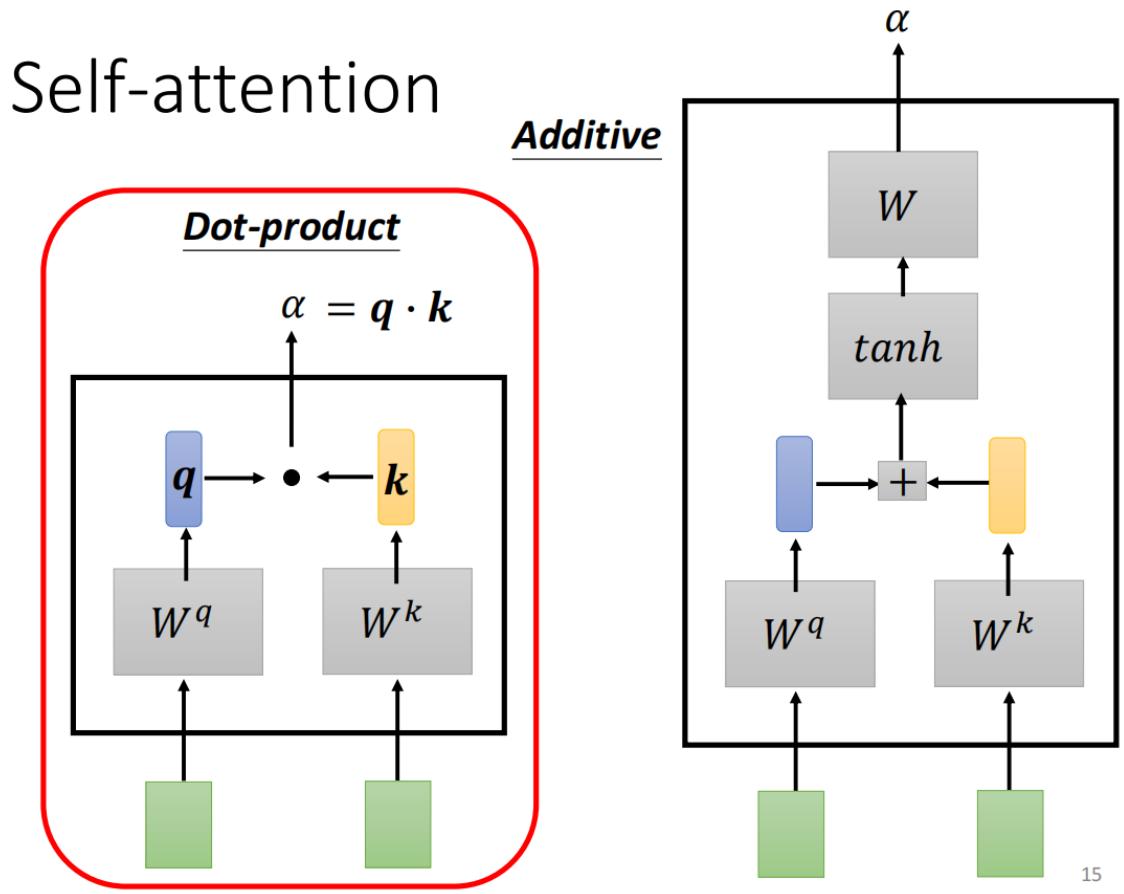
- 每个输入向量都有一个输出的标签: 比如词性标记
- 整个序列只有一个标签: 比如情感分析
- 机器自己决定输出的标签数量(seq2seq): 比如机器翻译

注意力机制

在分析一个句子的时候, 单独看一个单词的用处不大, 需要查看一整句话的意思才行. 因此需要分析上下文context信息, 找到句子中与单词相关的其他单词.



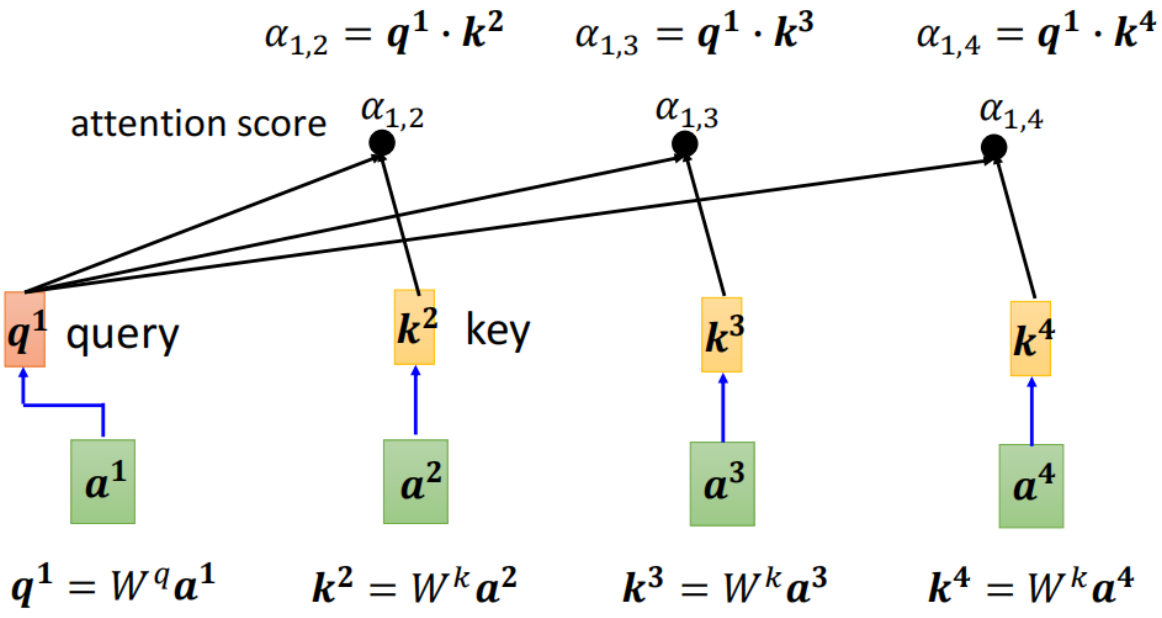
运算机制有两种, 点乘和加法



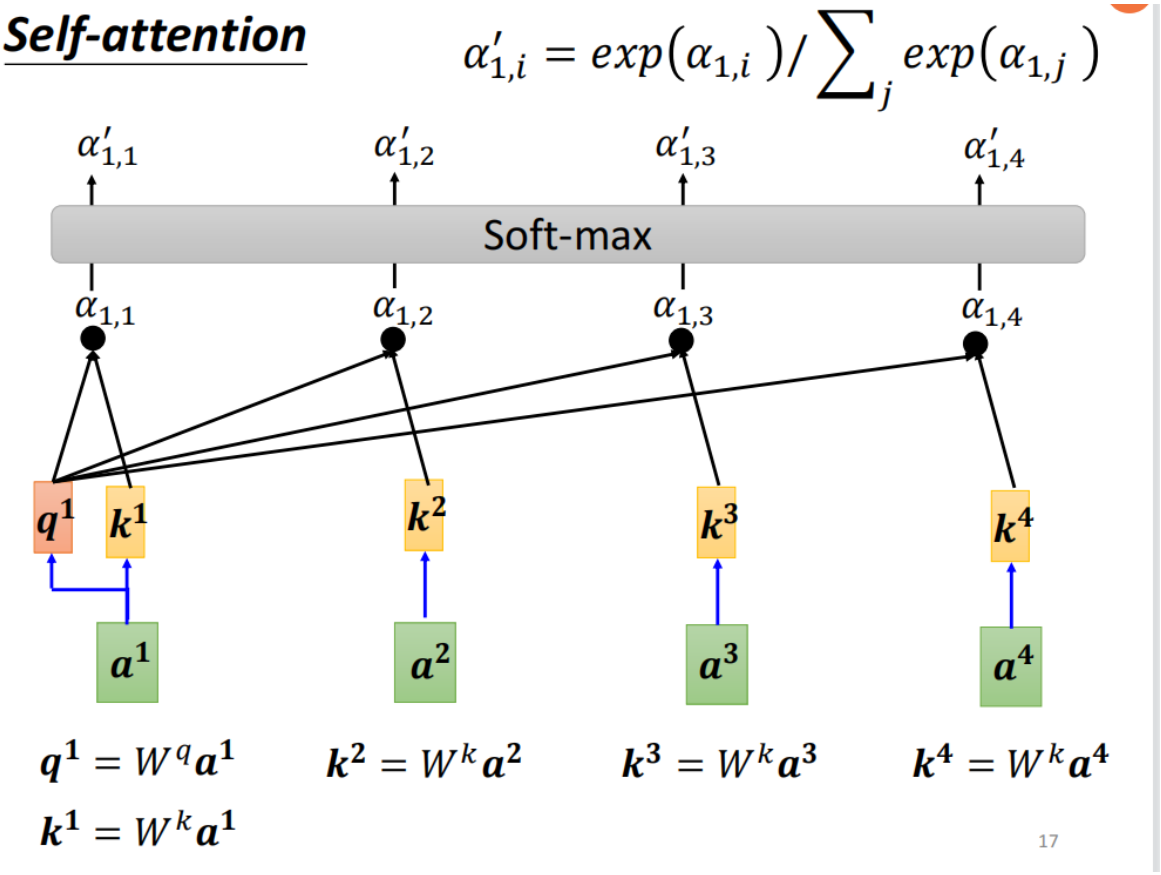
关键参数: W^Q , W^K, W^V

计算步骤

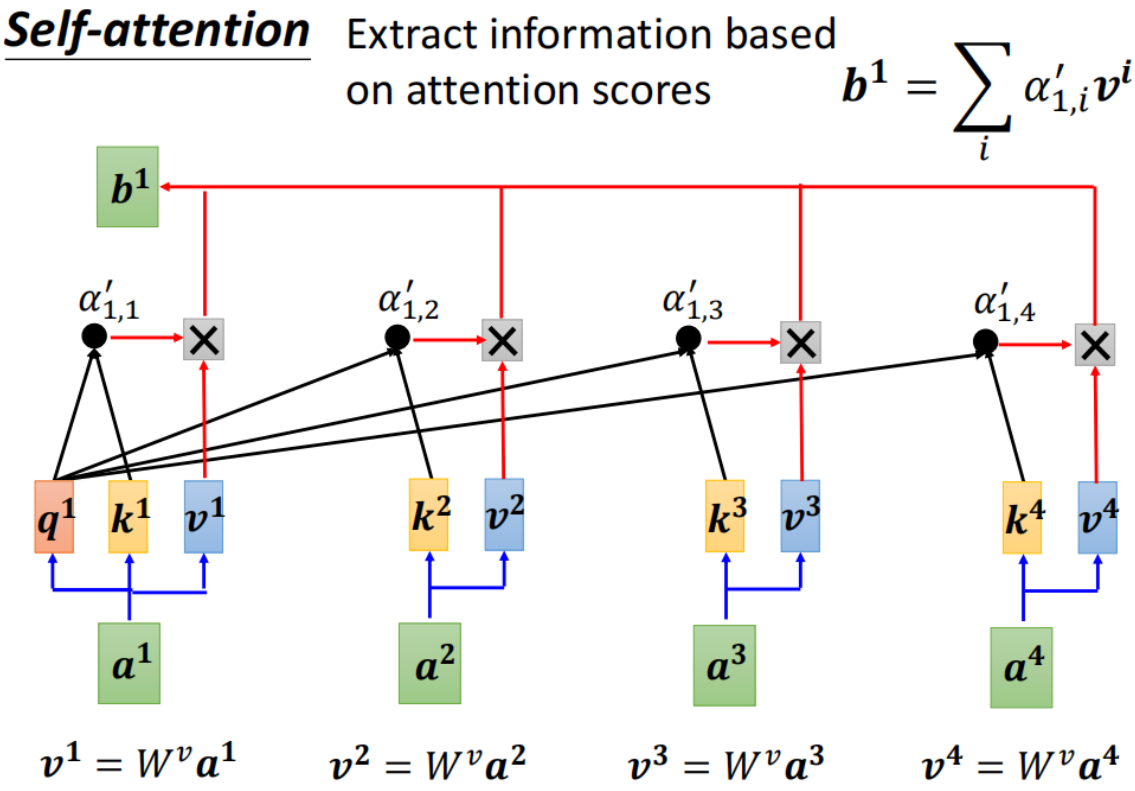
1. 计算W^q, W^k, 点乘得到attention score



2. 对attention score进行激活, 使用softmax或sigmoid



3. 计算W^v, 进行权重求和输出结果



白话版理解

为什么要因为注意力机制

在Attention诞生之前，已经有CNN和RNN及其变体模型了，那为什么还要引入attention机制？主要有两个方面的原因，如下：

- **计算能力的限制**：当要记住很多“信息“，模型就要变得更复杂，然而目前计算能力依然是限制神经网络发展的瓶颈。
- **优化算法的限制**：LSTM只能在一定程度上缓解RNN中的长距离依赖问题，且信息“记忆”能力并不高。

注意力机制

从本质上理解，Attention是从大量信息中有筛选出少量重要信息，并聚焦到这些重要信息上，忽略大多不重要的信息。权重(attention score)越大越聚焦于其对应的Value值上，即权重代表了信息的重要性，而Value是其对应的信息。

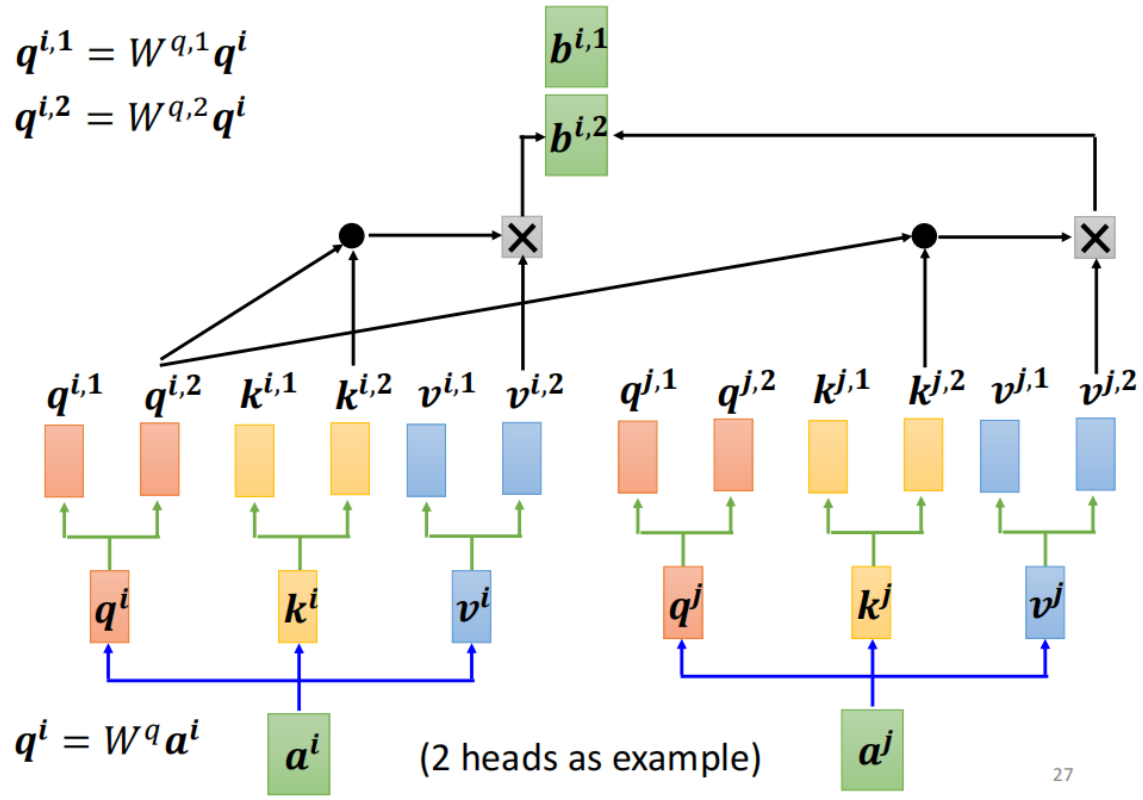
1. 将一个句子(sequence)的每个单词进行word embedding转化为向量, 并将这多个向量作为模型的输入;
2. 每个向量会分别乘以三个向量W^Q, W^K, W^V得到q^i, k^i, v^i;
3. 将q^i与所有的k^i进行点乘, 得到该个词向量与其他词向量的相关度 a
4. 对a进行softMax归一化得到$a^{'}$, 即注意力分数attention score(或者可看成是权重)
5. 将$a_i^{'}$分别与对应的v^i进行相乘累加, 得到最终的输出值

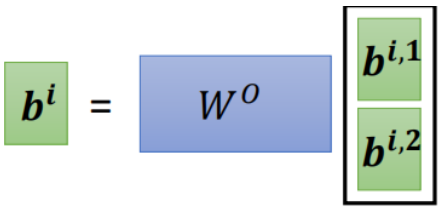
self-attention的输入既可以是模型的输入, 也可以是上一层self-attention的输出

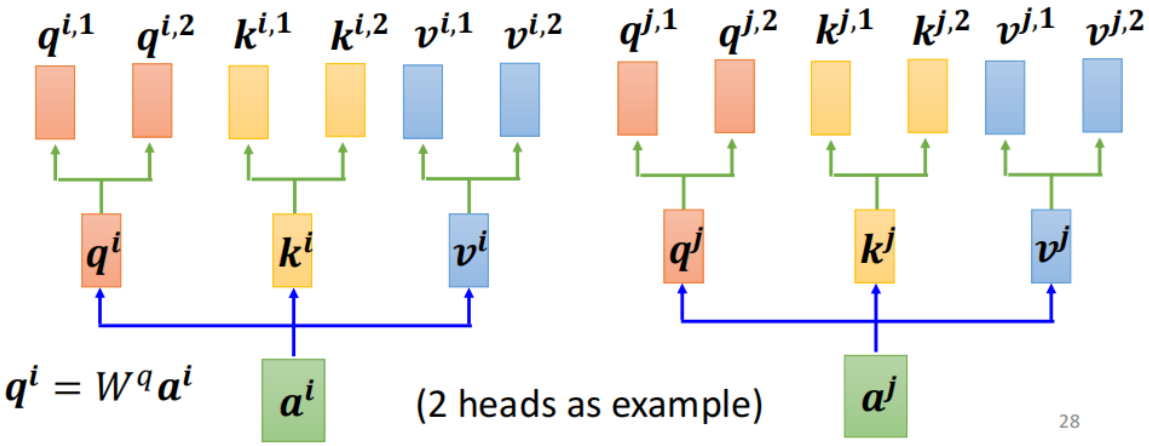
变种

多头注意力机制(multi-head self-attention)

- 以2头为例:

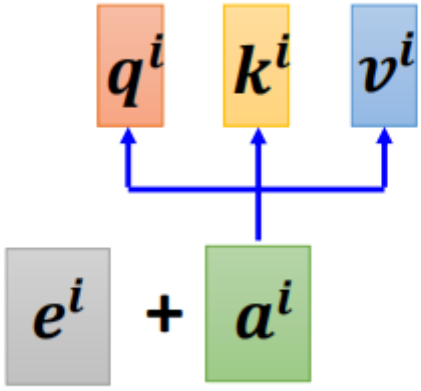






位置性编码Positional Encoding

在输入的向量中加入位置信息e^i



self-attention v.s. CNN

- self-attention的灵活性比CNN的更高, 即解空间更大(因此需要更多的数据)
- CNN只关注感受野里面其他向量的信息, self-attention关注全局向量的信息
- 因此只要做合适的配置, self-attention也可以转换成CNN

self-attention v.s. RNN

- RNN只能串行计算, self-attention可以并行计算
- RNN存在长距离依赖问题, self-attention解决了次问题