Self-attention

ATTENTION

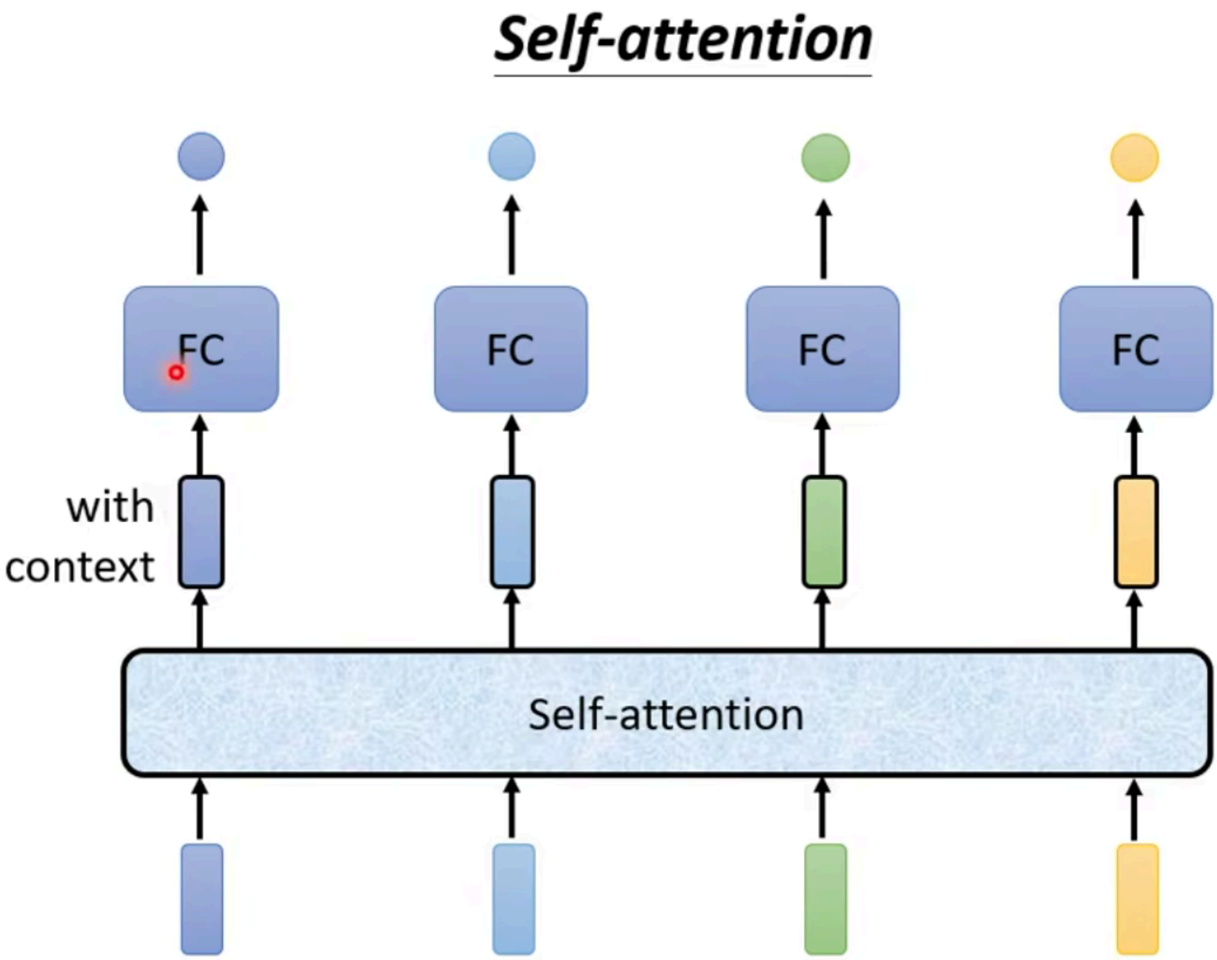
1. 输入与输出

- **输入：** 一堆向量
- **输出：**

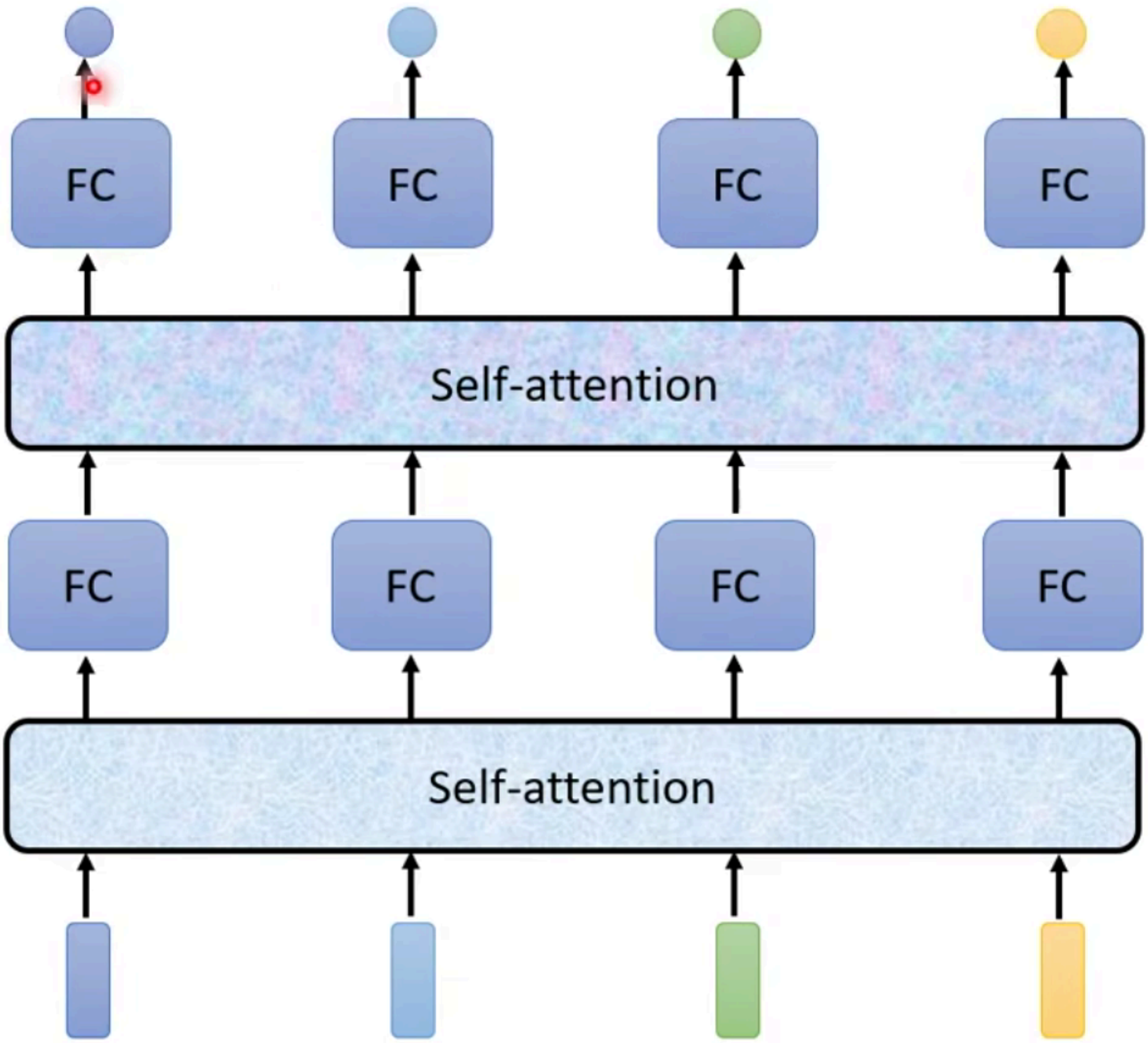
1. 每一个向量都有一个label（输入输出数目一样->Sequence labeling）
2. 整个序列只有一个label
3. 模型自己决定了输出的label的数量（seq2seq）

2. Sequence Labeling

2.1 机制



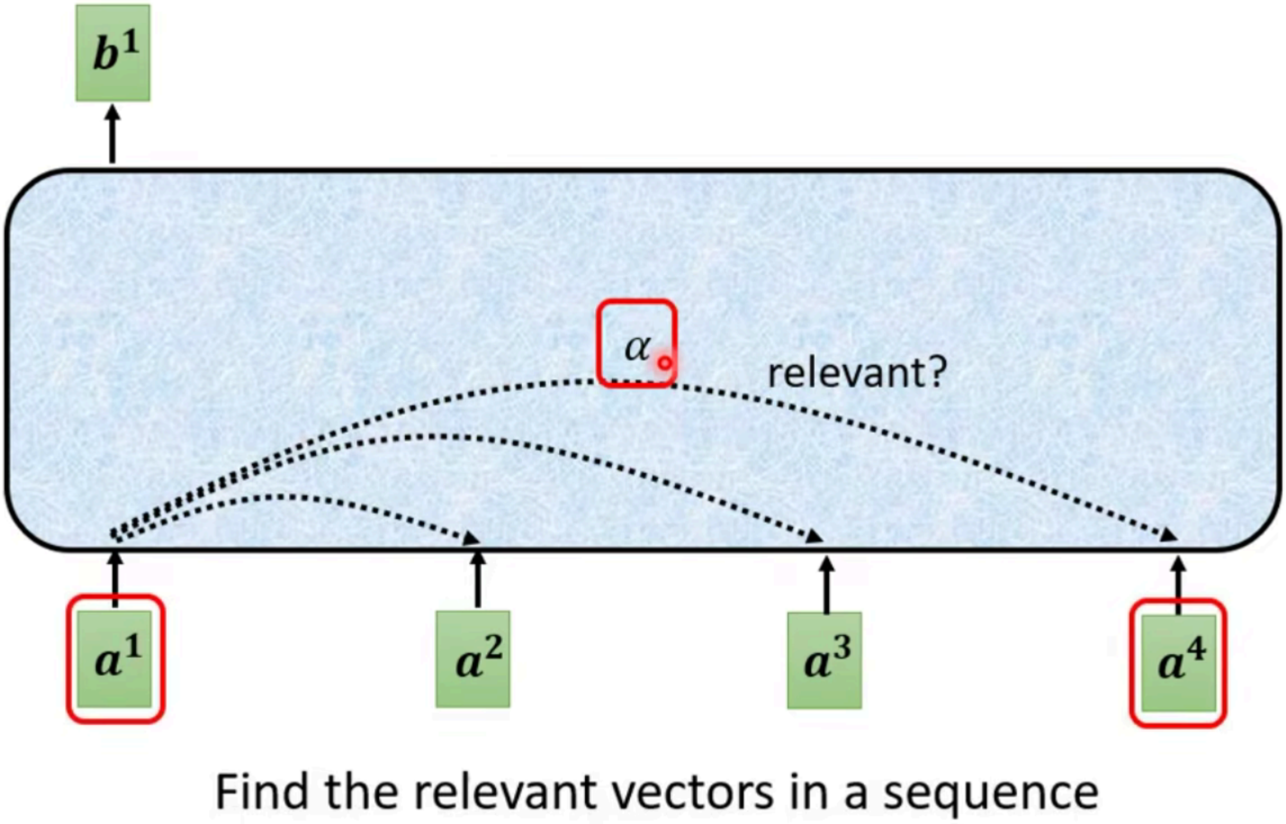
self-attention 处理的是整个序列的信息，得到考虑一整个序列的信息；FC(Fully Connected Network) 处理的是局部的信息。

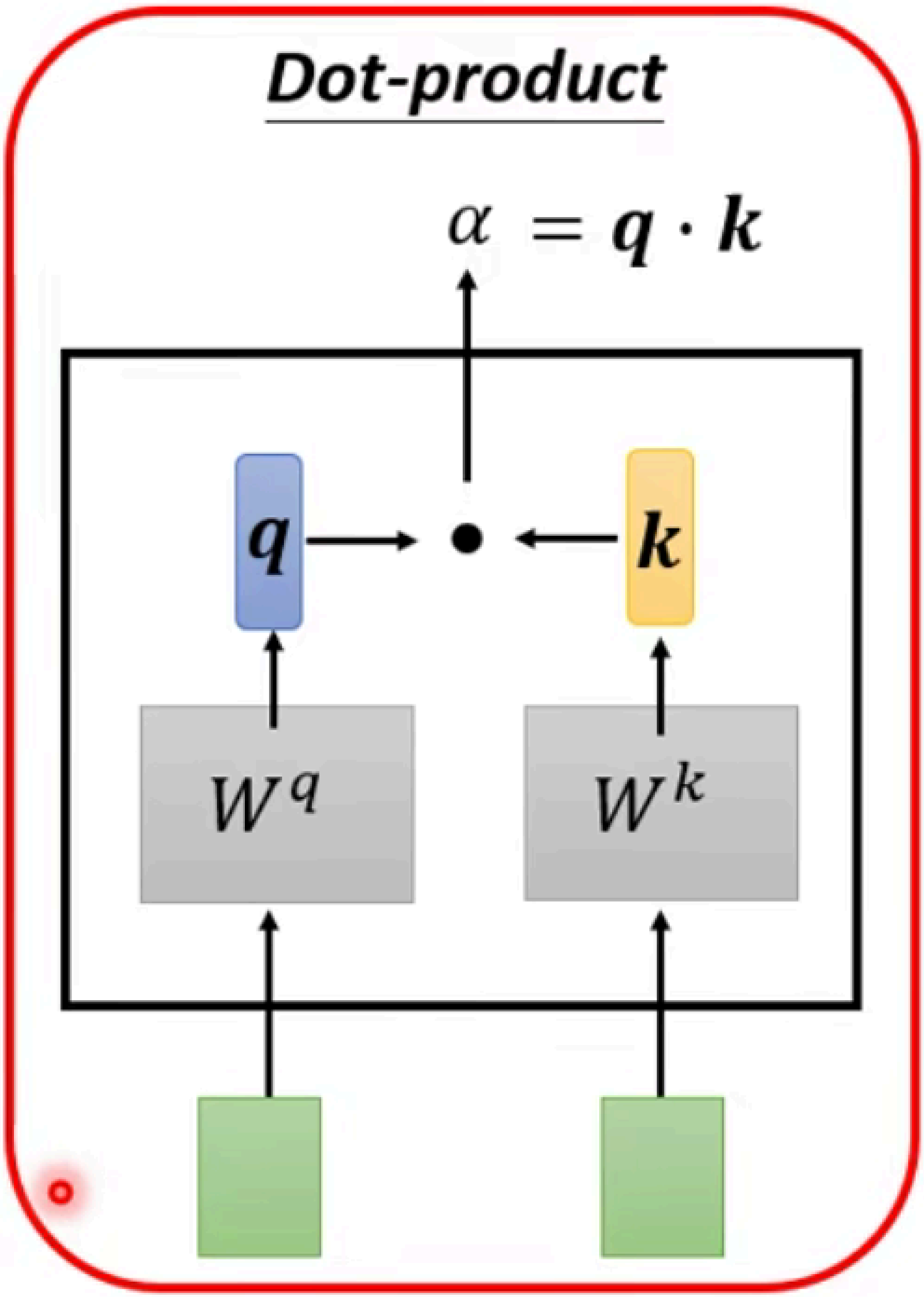


self-attention可以不断叠加，与FC交替出现

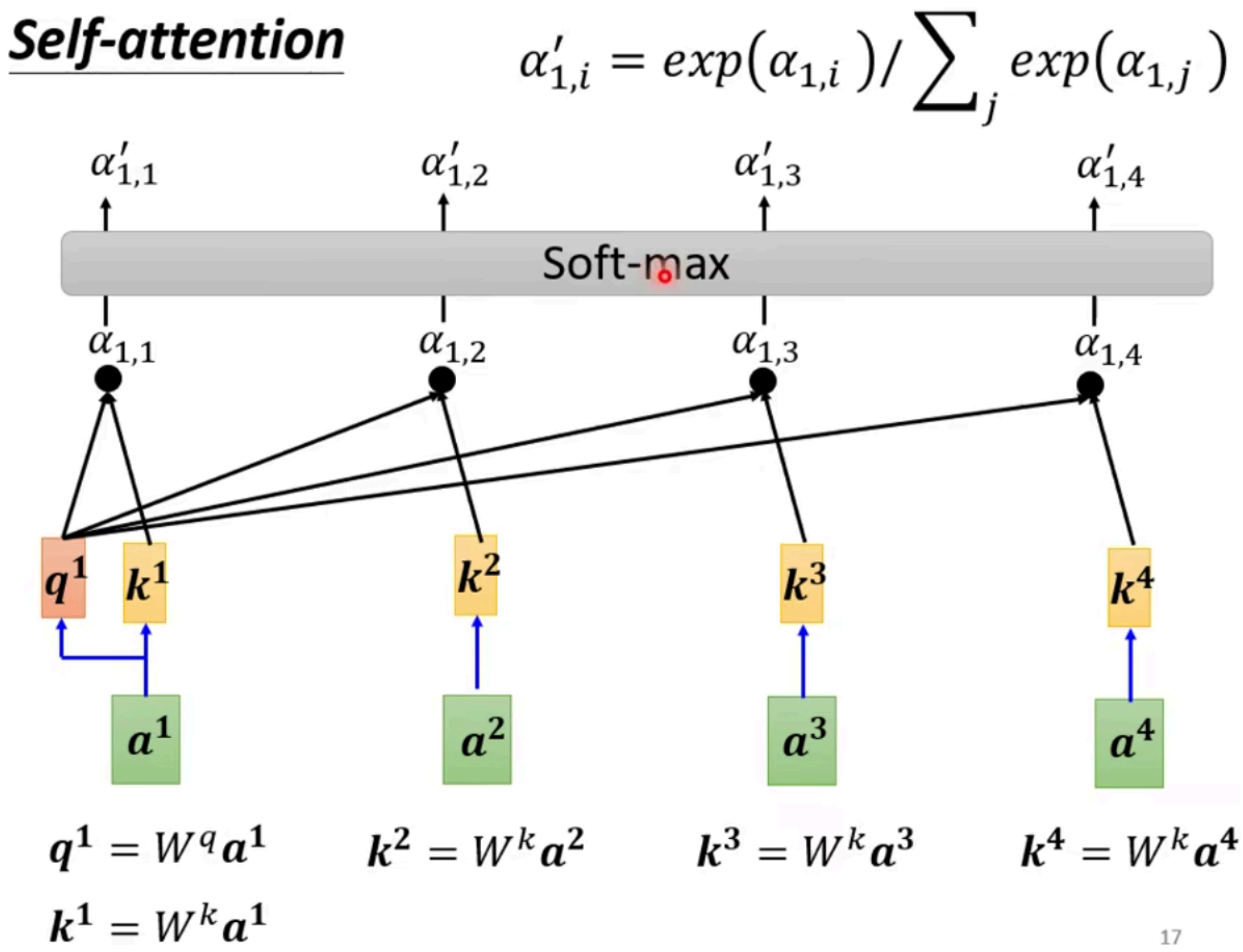
2.2 产生的步骤

1. 找出在序列中的相关的向量，用α代表两个向量的关联性



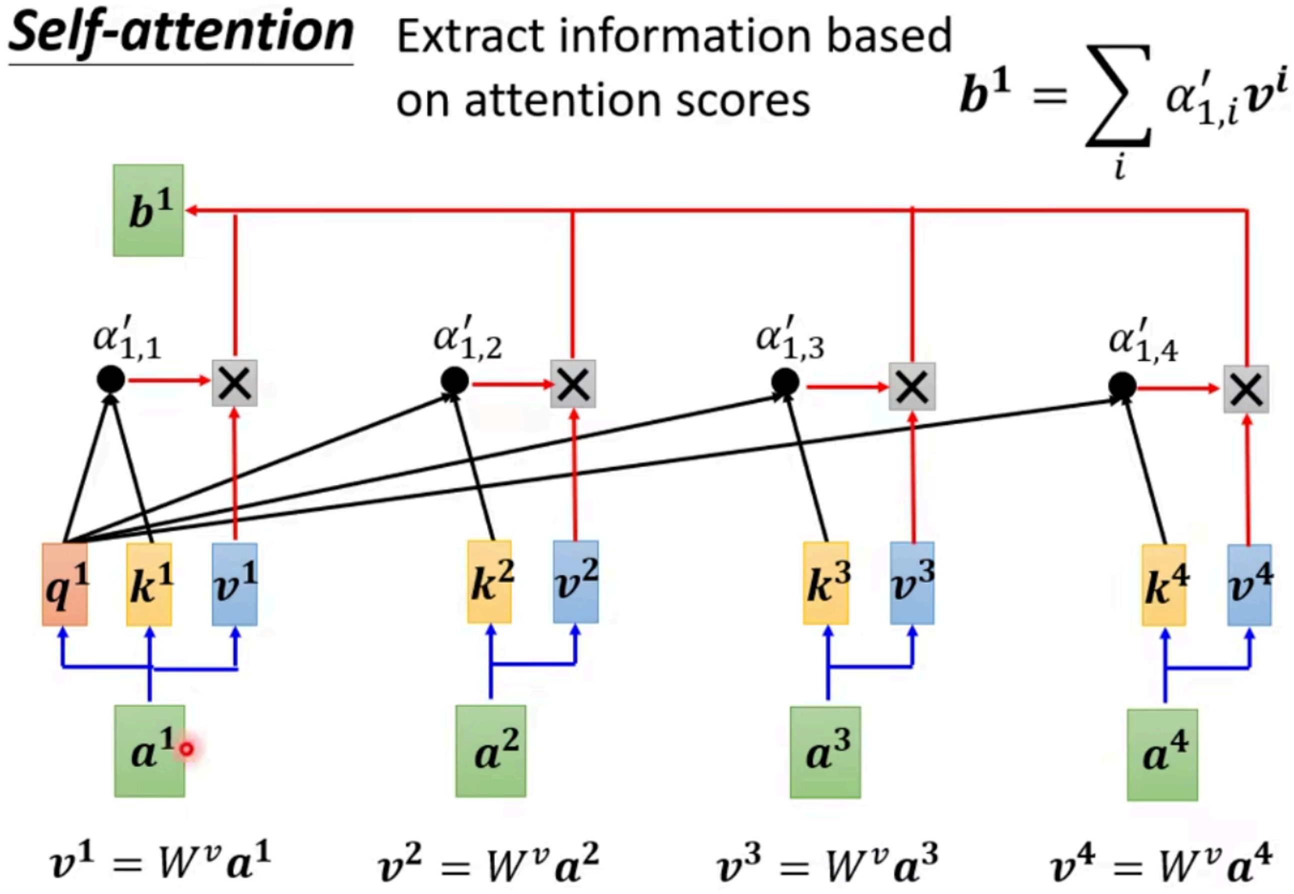


Dot-product
$\alpha = \boldsymbol{q} \cdot \boldsymbol{k}$
$\boldsymbol{q}$
$\boldsymbol{k}$
W^q
W^k



α1左乘Wq，剩余的阿尔法左乘Wk，分别得到q1与k2，k3，k4……一般来说α1也需要左乘Wq，得到k1。将q1与k1、k2、k3、k4……点乘可以得到相关系数α1,1 α1,2 α1,3（可以叫做attention score）…… 还需要通过Soft-max机制，转换相关系数

2. 根据α1,i'抽取出序列中重要的信息



将αi左乘Wv，可以得到vi，vi与α1,i'相乘，再将每一个乘积相加得到b1，vi越大，越接近抽取出来的结果b1

3. 矩阵表示

$$Q = W^q * I$$

$$K = W^k * I$$

$$V = W^v * I$$

$$Q = [q^1, q^2, q^3, q^4], K = [k^1, k^2, k^3, k^4],$$

$$V = [v^1, v^2, v^3, v^4], I = [\alpha^1, \alpha^2, \alpha^3, \alpha^4]$$

$$\begin{bmatrix} \alpha_{1,1} & \alpha_{2,1} & \alpha_{3,1} & \alpha_{4,1} \\ \alpha_{1,2} & \alpha_{2,2} & \alpha_{3,2} & \alpha_{4,2} \\ \alpha_{1,3} & \alpha_{2,3} & \alpha_{3,3} & \alpha_{4,3} \\ \alpha_{1,4} & \alpha_{2,4} & \alpha_{3,4} & \alpha_{4,4} \end{bmatrix} = \begin{bmatrix} k_1^T \\ k_2^T \\ k_3^T \\ k_4^T \end{bmatrix} * \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}$$

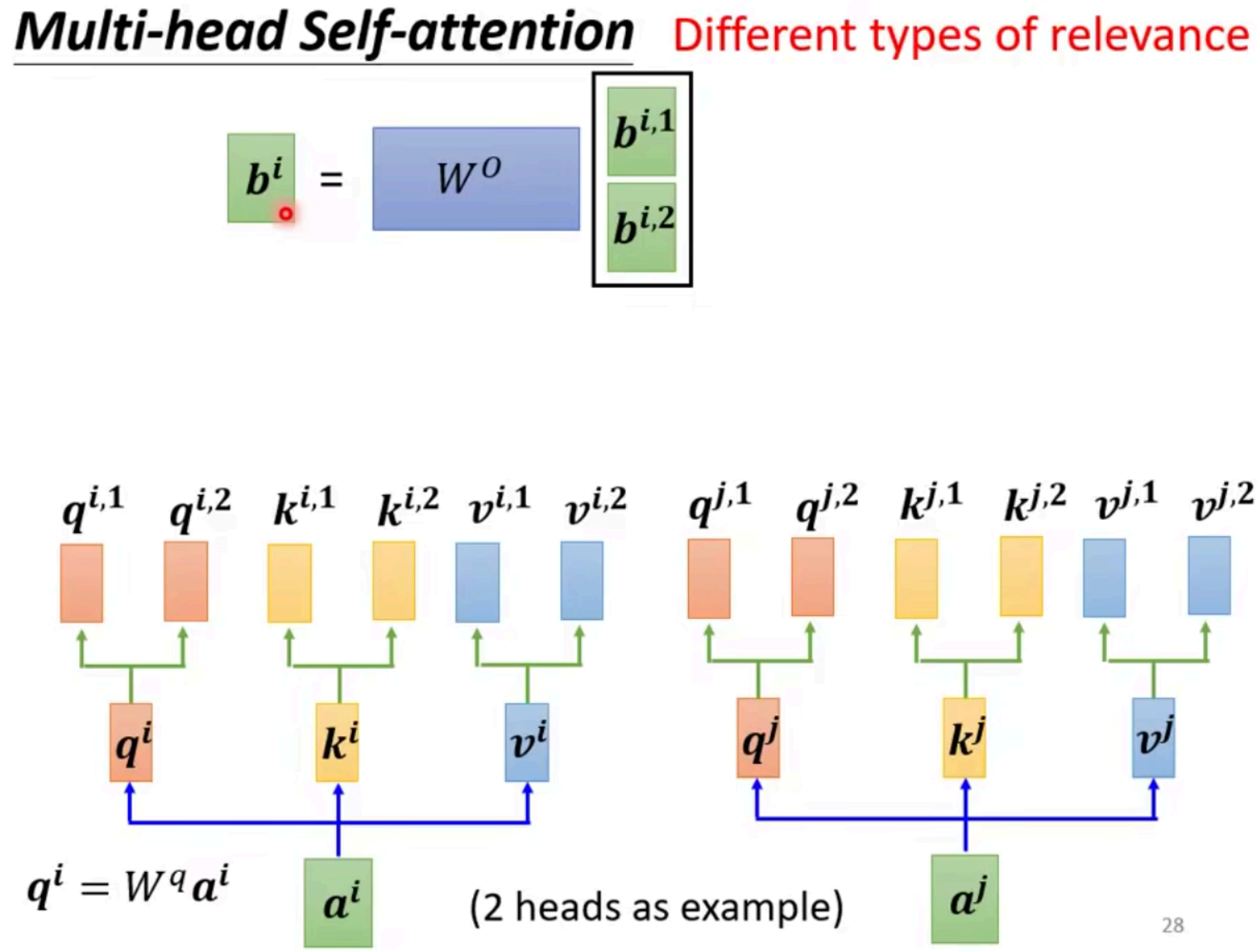
$$A = K^T * Q$$

$$A \to A'$$

$$O = V * A'$$

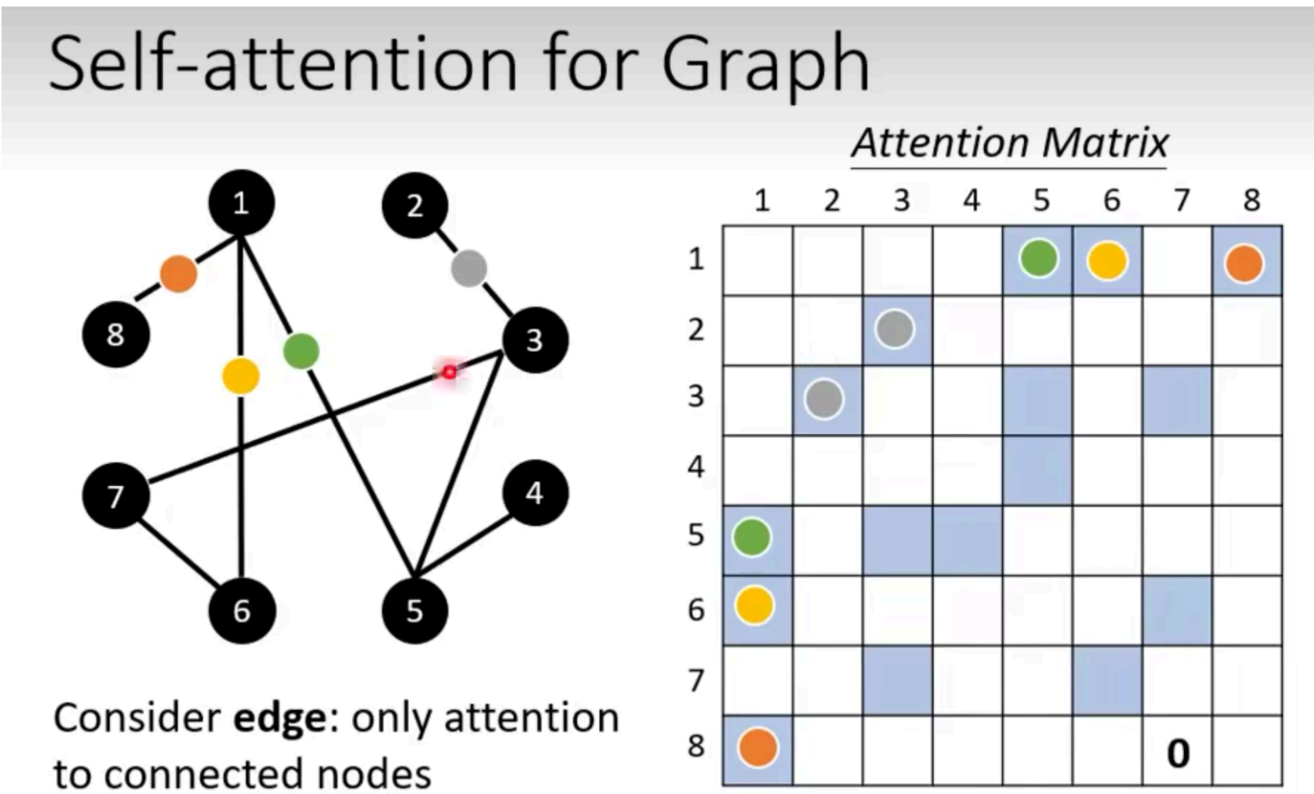
Wq,Wk,Wv需要通过学习找出来

4. Multi-head Self-attention



但是注意到并没有考虑位置，因此引入e^i，用$e^i+\alpha^i$表示位置

5. 将self-attention应用于Graph



只需要计算有edge相连接的节点之间的相关系数

进一步的学习

- Long Range Arena: A Benchmark for Efficient Transformers https://arxiv.org/abs/2011.04006
- Efficient Transformers: A Survey https://arxiv.org/abs/2009.06732