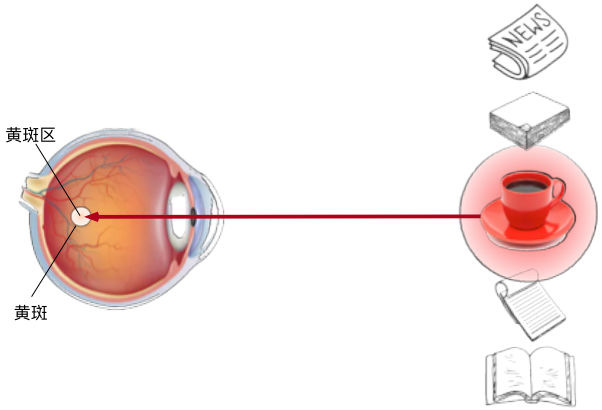
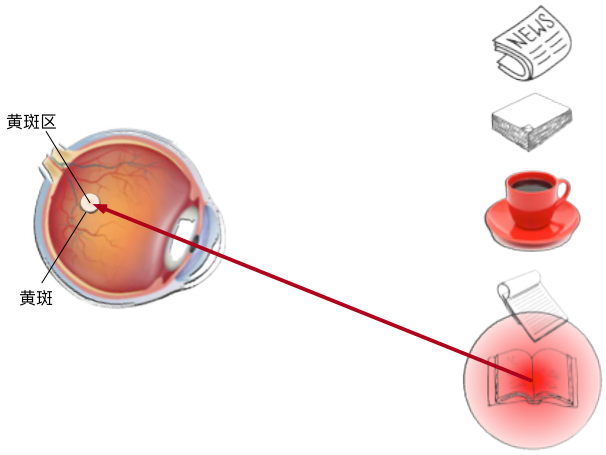
注意力机制

注意力机制是一个可以处理多个输入的技术。与传统的特征提取方式不同，注意力机制不会因为数据的显著特征而对其产生关注，而是通过给不同的数据赋予不同的权重，从而使模型关注于当前任务最相关的信息。

不随意线索：被动关注具有显著特征的数据



随意线索：主动关注具有主观意愿的数据

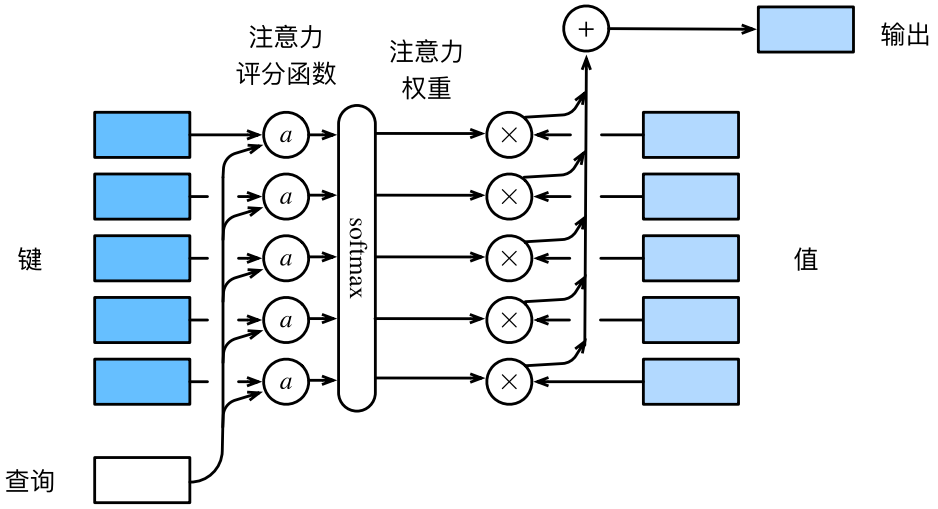


注意力汇聚

注意力机制通过三个元素**(query, key, value)**对数据进行加权，从而实现对相关信息产生关注，其中：

- **Query（查询）**：自主性提示，随意线索。
- **Key（键）**：非自主性提示，不随意线索。
- **Value（值）**：感官输入，每一个值有一个对应的键。

在注意力汇聚中，使用**查询**和**键**匹配在一起，实现对**值**的选择倾向。



计算过程

注意力汇聚计算公式如下：

$$f(x) = \sum_{i=1}^{n} \alpha(x, x_i) y_i \tag{1}$$

其中，$\alpha(x, x_i)$为注意力权重，y_i为i元素对应的值，$f(x)$为注意力汇聚的结果。

注意力权重计算公式如下：

$$\alpha(x, x_i) = softmax(a(x, x_i)) \tag{2}$$

其中$a(x, x_i)$为**注意力分数**，代表两个元素的**相似程度**，通常可使用查询和键的相似度表示，也可写为$a(q, k_i)$。

计算注意力权重的过程是计算目标数据查询与所有输入数据的键的**匹配程度**作为注意力分数，虽有对所有的注意力分数进行**softmax函数**处理，得到该元素应该获取的注意力权重。

注意力分数有两种计算方式：

- **加性注意力**（主要针对Q, K, V维度不相同的情况）

$$a(q, k) = \mathbf{W}_v^\top tanh(\mathbf{W}_q q + \mathbf{W}_k k) \in \mathbb{R} \tag{3}$$

  其中$\mathbf{W}_q \in \mathbb{R}^{h \times q}$， $\mathbf{W}_k \in \mathbb{R}^{h \times k}$， $\mathbf{W}_v \in \mathbb{R}^h$。这些矩阵的意义是将**key**和**query**映射到**value**的同一维度空间，然后使用非线性激活函数$tanh$，最后通过一个权重向量$W_v^\top$进行加权求和来计算得分。加性指的是将**key**和**query**加到一起。

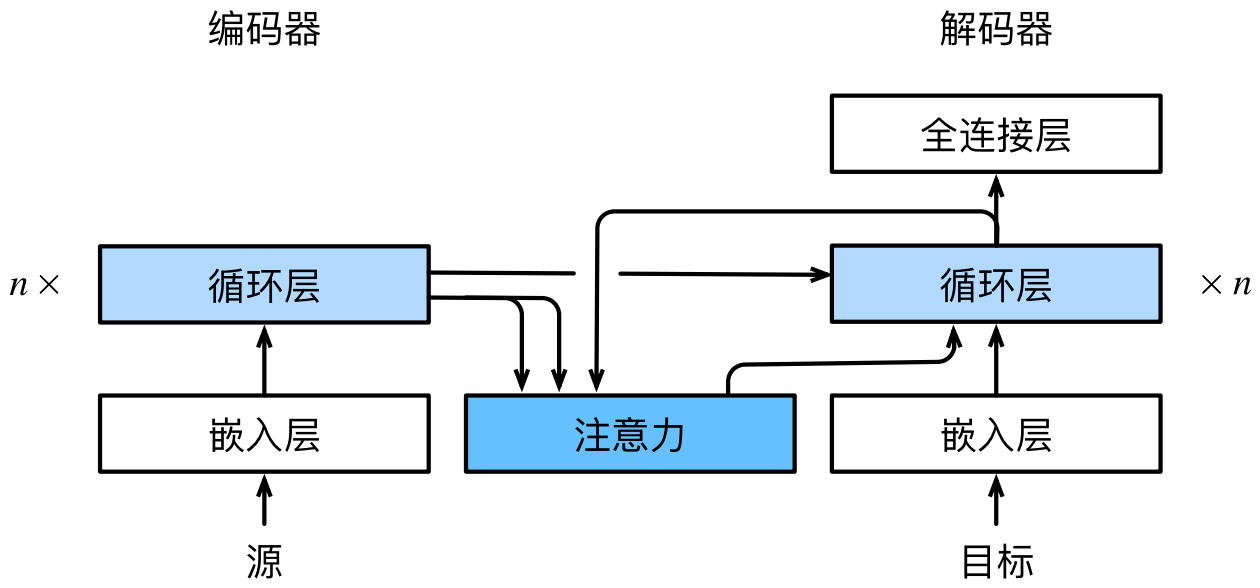
- **Scaled Dot-product Attention** （主要应对Q, K相同维度的情况，即自注意力）

$$a(q, k) = \frac{q \top k}{\sqrt{d}} \tag{4}$$

  其中$q, k \in \mathbb{R}^d$。由于**key**和**query**有相同的维度，可以直接点乘得到数值，再通过$\sqrt{d}$元素来去除特征向量长度的影响。

使用注意力的seq2seq

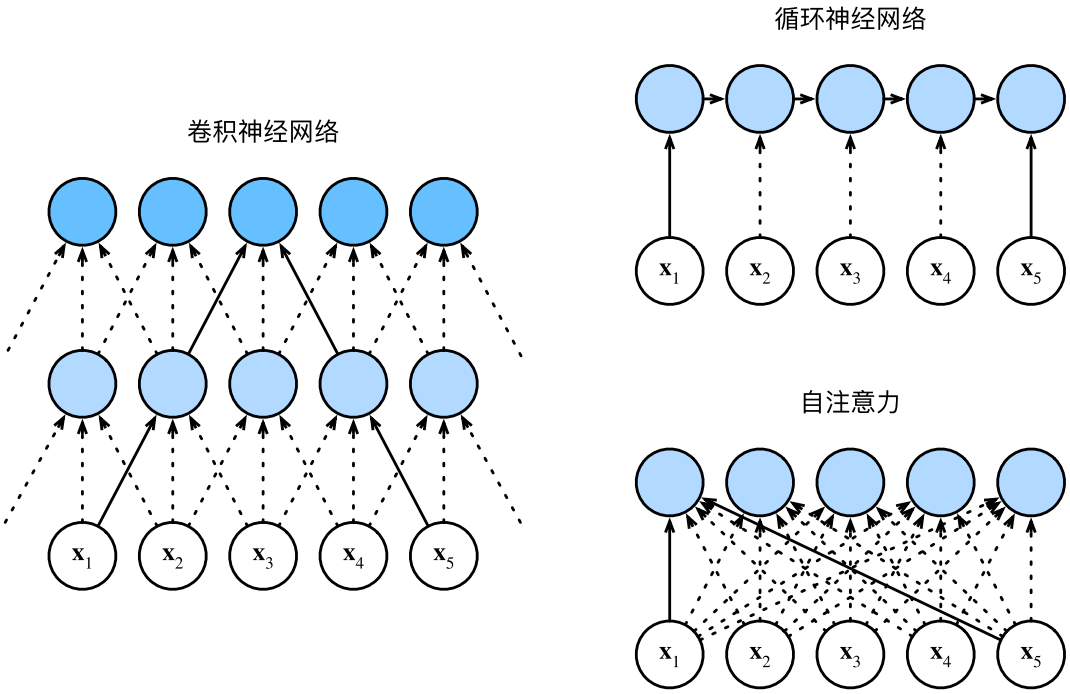
在**RNN**的$seq2seq$模型中加入注意力机制，可以使模型在翻译时可以考虑输入元素在序列中的相对位置。



- 编码器在所有时间步的最终层隐状态作为注意力机制的**key**和**value**。
- 编码器的最终状态作为解码器的初始化隐状态。
- 解码时，上一步的最终层隐状态作为**query**。
- 注意力输出和embedding作为输入。

自注意力

自注意力使用用一组词元同时充当**query**, **key**和**value**。



位置编码

位置编码是将位置信息编码，并作为输入的一部分。这样做的目的是为了使注意力机制可以考虑位置信息，计算方式如下：

$$X \in \mathbb{R}^{n\times d}, P \in \mathbb{R}^{n\times d}, \text{input} = P + X \tag{5}$$

其中P为编码的位置信息，给每一个**元素**的每一个**feature**进行编码，其中偶数位的feature使用sin函数，奇数位的feature使用cos函数，不同的元素具有不同的频率，具体方式如下：

$$\begin{aligned} P_{i,2j} &= sin(\frac{i}{10000^{2j/d}}), \\ P_{i,2j+1} &= cos(\frac{i}{10000^{2j/d}}). \end{aligned} \tag{6}$$

由于三角函数具有性质：

$$\begin{cases} sin(\alpha+\beta) = sin(\alpha)cos(\beta) + sin(\beta)cos(\alpha) \\ cos(\alpha+\beta) = cos(\alpha)cos(\beta) - sin(\alpha)sin(\beta) \end{cases} \tag{7}$$

所以位置编码可以将元素**相对位置**的信息通过下面方式进行转换：

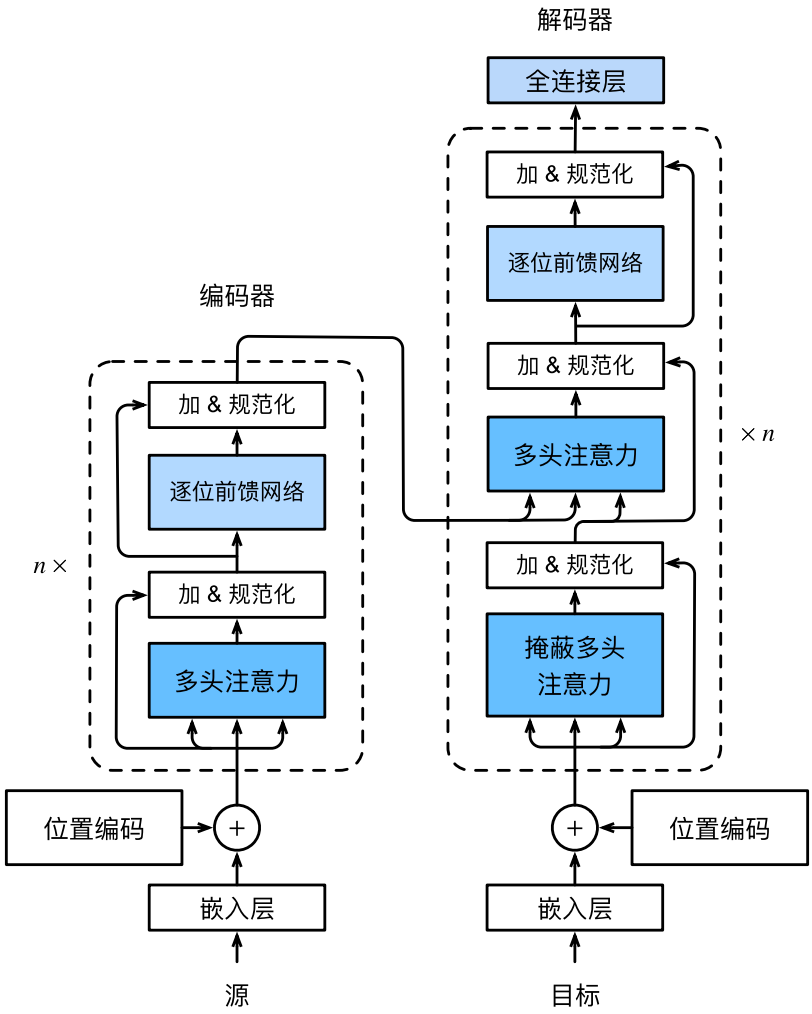
$$\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

$$
\begin{aligned}
&\begin{bmatrix} \cos(\delta\omega_j) & \sin(\delta\omega_j) \\ -\sin(\delta\omega_j) & \cos(\delta\omega_j) \end{bmatrix} \begin{bmatrix} p_{i,2j} \\ p_{i,2j+1} \end{bmatrix} \\
&= \begin{bmatrix} \cos(\delta\omega_j)\sin(i\omega_j) + \sin(\delta\omega_j)\cos(i\omega_j) \\ -\sin(\delta\omega_j)\sin(i\omega_j) + \cos(\delta\omega_j)\cos(i\omega_j) \end{bmatrix} \\
&= \begin{bmatrix} \sin\left((i+\delta)\omega_j\right) \\ \cos\left((i+\delta)\omega_j\right) \end{bmatrix} \\
&= \begin{bmatrix} p_{i+\delta,2j} \\ p_{i+\delta,2j+1} \end{bmatrix},
\end{aligned}
\tag{8}
$$

Transformer

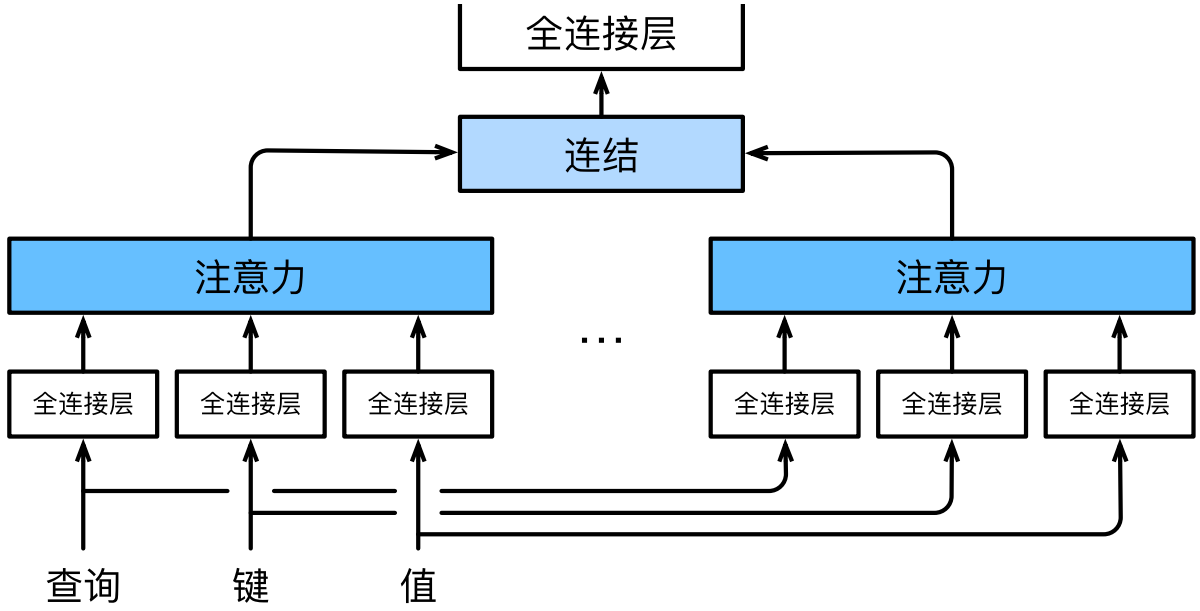
Transformer是一个由纯注意力机制组成的编码器解码器架构，可以理解为一种改进的$seq2seq$模型，将传统的**RNN**结构替换为基于注意力机制的结构。这种结构赋予了模型处理长序列的能力。

- 编码器和解码器中块的个数相同，输出维度相同
- 编译器中的输出$(y_1, y_2, \ldots, y_n)$作为解码器中第一个**transformer块**的**key**和**value**，目标序列元素作为**query**。
- 预测时，解码器的前t个预测作为**key**和**value**，第t个预测值作为**query**。根据已生成序列预测下一个词，直到生成特殊序列结束符号或达到最大长度限制。
- 在训练时，目标序列（真实序列）直接替代预测结果作为输入。



多头注意力

多头注意力指的是将**key**, **query**和**value**通过**线性变换**映射到多个不同的**线性空间**，在每个线性空间内**分别**进行注意力汇聚，最后将所有汇聚结果拼接在一起在进行一次**可学习的线性变换**得到结果。这样可以使模型学习到不同的行为，并将它们拼接起来。



以上过程可以表示为：

$$\begin{aligned}\mathbf{h}_i &= f(\mathbf{W}_i^{(q)}\mathbf{q}, \mathbf{W}_i^{(k)}\mathbf{k}, \mathbf{W}_i^{(v)}\mathbf{v}) \in \mathbb{R}^{p_v}, \\ \mathbf{W}_o &\begin{bmatrix}\mathbf{h}_1 \\ \vdots \\ \mathbf{h}_h\end{bmatrix} \in \mathbb{R}^{p_o}.\end{aligned} \tag{9}$$