具体一句话（比如 **“I have a cat”**）是如何一步步转化为向量，然后通过 **Q、K、V** 做 self-attention 计算的。

一、输入序列

句子是：

> “I have a cat”

长度 = 4
词表中我们有每个词的 **embedding**（词向量）。
比如假设词向量维度是 ($d_{model} = 4$)。

词	原始 embedding 向量	
I	[0.2, 0.4, 0.1, 0.5]	
have	[0.9, 0.1, 0.3, 0.7]	
a	[0.5, 0.2, 0.4, 0.1]	
cat	[0.6, 0.9, 0.8, 0.3]	

这些是模型的输入（Embedding 层 + 位置编码后得到的输入矩阵）：

$$X = [0.2 \quad 0.4 \quad 0.1 \quad 0.5\, 0.9 \quad 0.1 \quad 0.3 \quad 0.7\, 0.5 \quad 0.2 \quad 0.4 \quad 0.1\, 0.6 \quad 0.9 \quad 0.8 \quad 0.3]$$

维度：

$$X \in \mathbb{R}^{4\times 4} \quad (4\text{个词，每个词是}4\text{维向量})$$

二、生成 Q、K、V

Transformer 中我们定义三个可学习的权重矩阵：

$$W_Q, W_K, W_V \in \mathbb{R}^{4\times 4}$$

比如假设它们是：

$$W_Q = [0.1 \quad 0.2 \quad 0.3 \quad 0.4\, 0.5 \quad 0.6 \quad 0.7 \quad 0.8\, 0.9 \quad 0.1 \quad 0.2 \quad 0.3\, 0.4 \quad 0.5 \quad 0.6 \quad 0.7],$$

$$W_K = [0.2 \quad 0.3 \quad 0.4 \quad 0.1\, 0.5 \quad 0.7 \quad 0.8 \quad 0.6\, 0.9 \quad 0.1 \quad 0.3 \quad 0.5\, 0.4 \quad 0.2 \quad 0.9 \quad 0.7],$$

$$W_V = [0.1 \quad 0.3 \quad 0.5 \quad 0.7\, 0.2 \quad 0.4 \quad 0.6 \quad 0.8\, 0.9 \quad 0.1 \quad 0.2 \quad 0.3\, 0.5 \quad 0.6 \quad 0.7 \quad 0.8]$$

然后通过线性变换：

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V$$

这会给我们三个矩阵：

矩阵	形状	含义
Q	(4×4)	每个词的“查询”向量
K	(4×4)	每个词的“键”向量
V	(4×4)	每个词的“值”向量

三、计算注意力分数（QK^T）

我们用点积来计算“第 i 个词关注第 j 个词的程度”：

$$score(i,j) = Q_i \cdot K_j$$

例如：

- “I” 的 Query 向量与 “have” 的 Key 向量点积 = 它们的相似程度；
- “I” 的 Query 与 “cat” 的 Key 向量点积 = 是否语义相关。

最终我们得到一个 **4×4 的注意力分数矩阵**：

$$S = QK^T$$

每一行代表“当前词”，每一列代表“被关注的词”。

四、Softmax 归一化（得到权重）

对每一行做 softmax，使得每个词对所有词的注意力权重之和为 1：

$$A = softmax\left(\frac{S}{\sqrt{d_k}}\right)$$

(A_{ij}) 就是：

> 第 i 个词关注第 j 个词的程度。

例如（举个假设的结果）：

Query/Key	I	have	a	cat
I	0.10	0.50	0.20	0.20
have	0.25	0.25	0.25	0.25
a	0.15	0.25	0.30	0.30
cat	0.05	0.15	0.20	0.60

这里可以看到：

- “cat” 主要关注自己（0.60）和 “a”（0.20）
- “I” 比较关注 “have” （0.5）

五、加权求和 Value 向量（得到输出）

最后一步：

$$Z = AV$$

也就是：
每个词的新表示(z_i)是对所有 Value 向量 (V_j) 的加权平均。

举个直觉：

- *\\$I$\\"的新表示 $= 0.10 \times V(I) + 0.50 \times V(have) + 0.20 \times V(a) + 0.20 \times V(cat)$
- \\$cat$\\"的新表示 $= 0.05 \times V(I) + 0.15 \times V(have) + 0.20 \times V(a) + 0.60 \times V(cat)$

这样，每个词都融合了上下文中其他词的信息（按重要性加权）。

六、最终输出

输出矩阵 (Z)（形状仍是 (4×4)）：

- 它和原始词向量维度一样；
- 但每个词的表示现在包含了“上下文依赖”的信息。