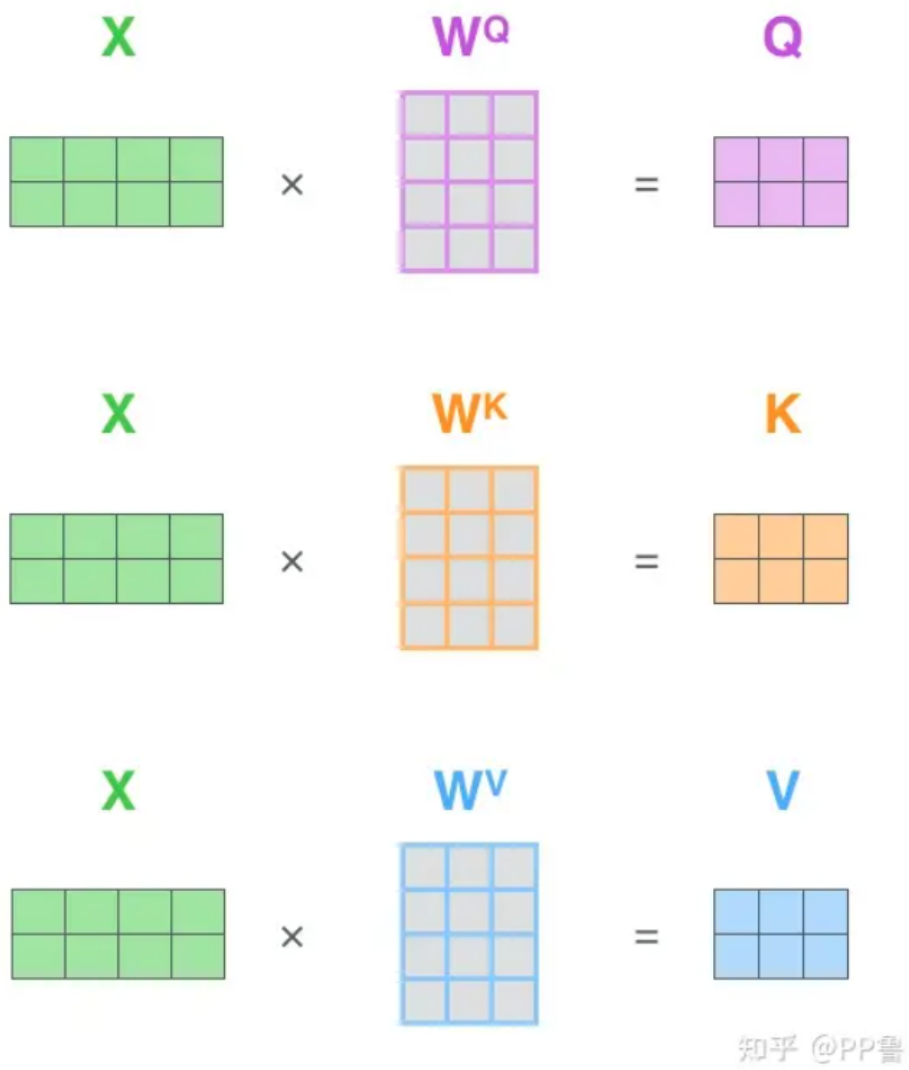
注意力Attention机制的最核心的公式为：$Softmax(\frac{QK^\top}{\sqrt{d_k}})V$，与我们刚才分析的$Softmax(\mathbf{X}\mathbf{X}^\top)\mathbf{X}$有几分相似。Transformer[^1]论文中将这个Attention公式描述为：Scaled Dot-Product Attention。其中，Q为Query、K为Key、V为Value。Q、K、V是从哪儿来的呢？Q、K、V其实都是从同样的输入矩阵X线性变换而来的。我们可以简单理解成：

$$
\begin{aligned}
Q &= XW^Q \\
K &= XW^K \\
V &= XW^V
\end{aligned}
$$

用图片演示为：

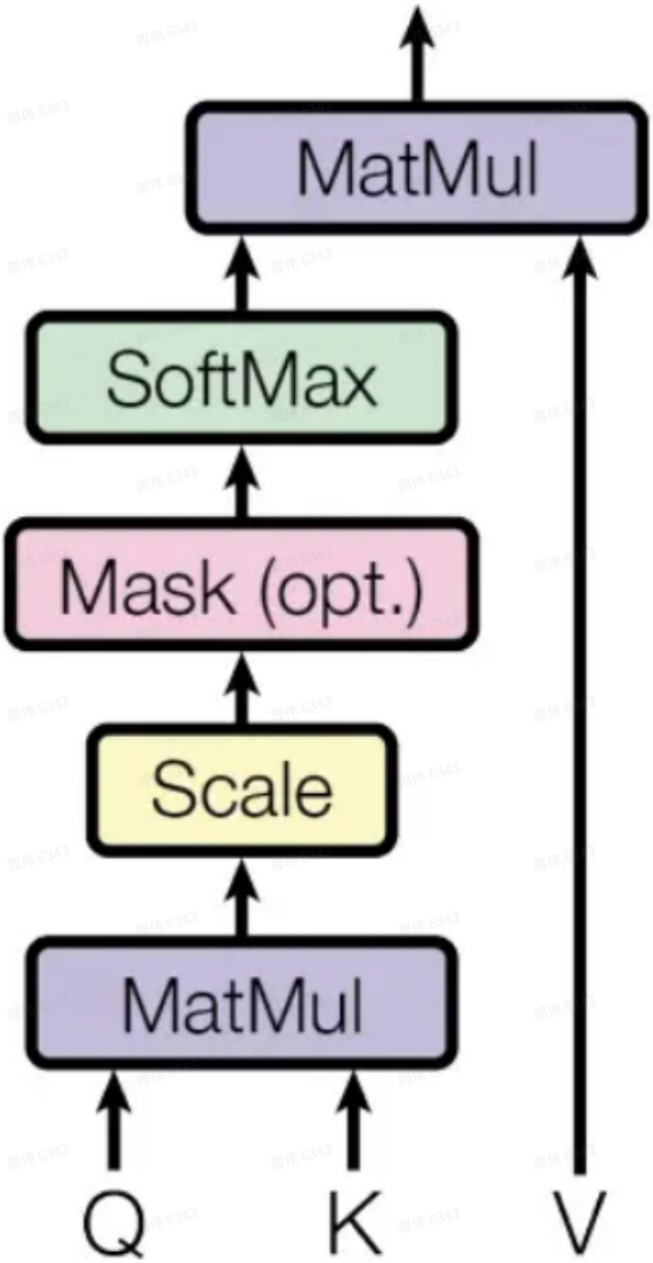


X分别乘以三个矩阵，生成Q、K、V矩阵

其中，W^Q，W^K和W^V是三个可训练的参数矩阵。输入矩阵X分别与W^Q，W^K和W^V相乘，生成Q、K和V，相当于经历了一次线性变换。Attention不直接使用X，而是使用经过矩阵乘法生成的这三个矩阵，因为使用三个可训练的参数矩阵，可增强模型的拟合能力。

Q与$K^\top$经过MatMul，生成了相似度矩阵。对相似度矩阵每个元素除以$\sqrt{d_k}$，d_k为K的维度大小。这个除法被称为Scale。当d_k很大时，$QK^\top$的乘法结果方差变大，进行Scale可以使方差变小，训练时梯度更新更稳定。

Mask是机器翻译等自然语言处理任务中经常使用的环节。在机器翻译等NLP场景中，每个样本句子的长短不同，对于句子结束之后的位置，无需参与相似度的计算，否则影响Softmax的计算结果。



简单图解Attention机制

第一步：输入为词向量矩阵X，每个词为矩阵中的一行，经过与W进行矩阵乘法，首先生成Q、K和V。`q1 = X1 * WQ`，`q1`为`Q`矩阵中的行向量，`k1`等与之类似。

第二步：进行$QK^\top$计算，得到相似度

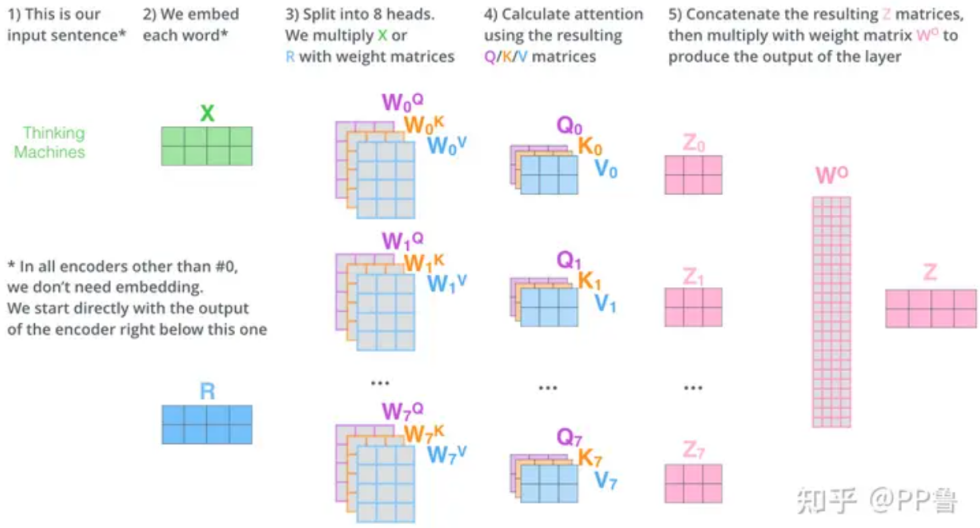
第三步：相似度除以$\sqrt{d_k}$，再进行Softmax。经过Softmax的归一化后，每个值是一个大于0小于1的权重系数，且总和为0，这个结果可以被理解成一个权重矩阵

第四步：使用刚得到的权重矩阵，与V相乘，计算加权求和

以上是注意力机制的一种解释，我觉得简单易懂，而且也很清晰，就直接抄下来了

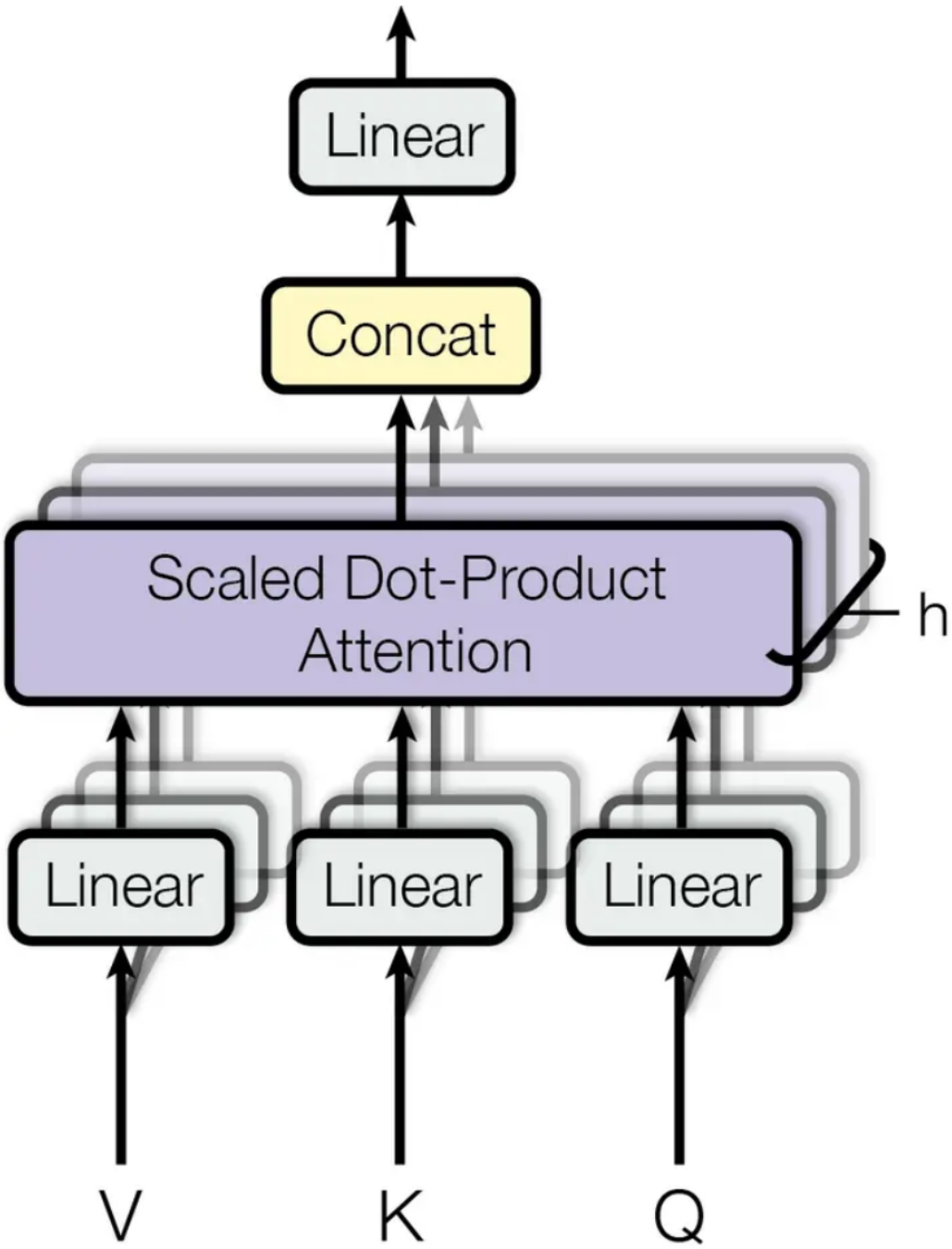
多头注意力：对于同样的输入X，我们定义多组不同的W^Q、W^K、W^V，比如W_0^Q、W_0^K、W_0^V，W_1^Q、W_1^K和W_1^V，每组分别计算生成不同的Q、K、V，最后学习到不同的参数。

相比于普通的注意力机制主要是在第一步，由于W^Q、W^K、W^V增多，所以得到的结果也增多，将结果得到的矩拼接起来，然后再进行降维，就能得到我们想要的输出



多头注意力计算过程

这张图比较形象，得到的Z0，Z1等等结果拼接起来，通过与W相乘来降维得到Z



大致过程应该就是这张图了