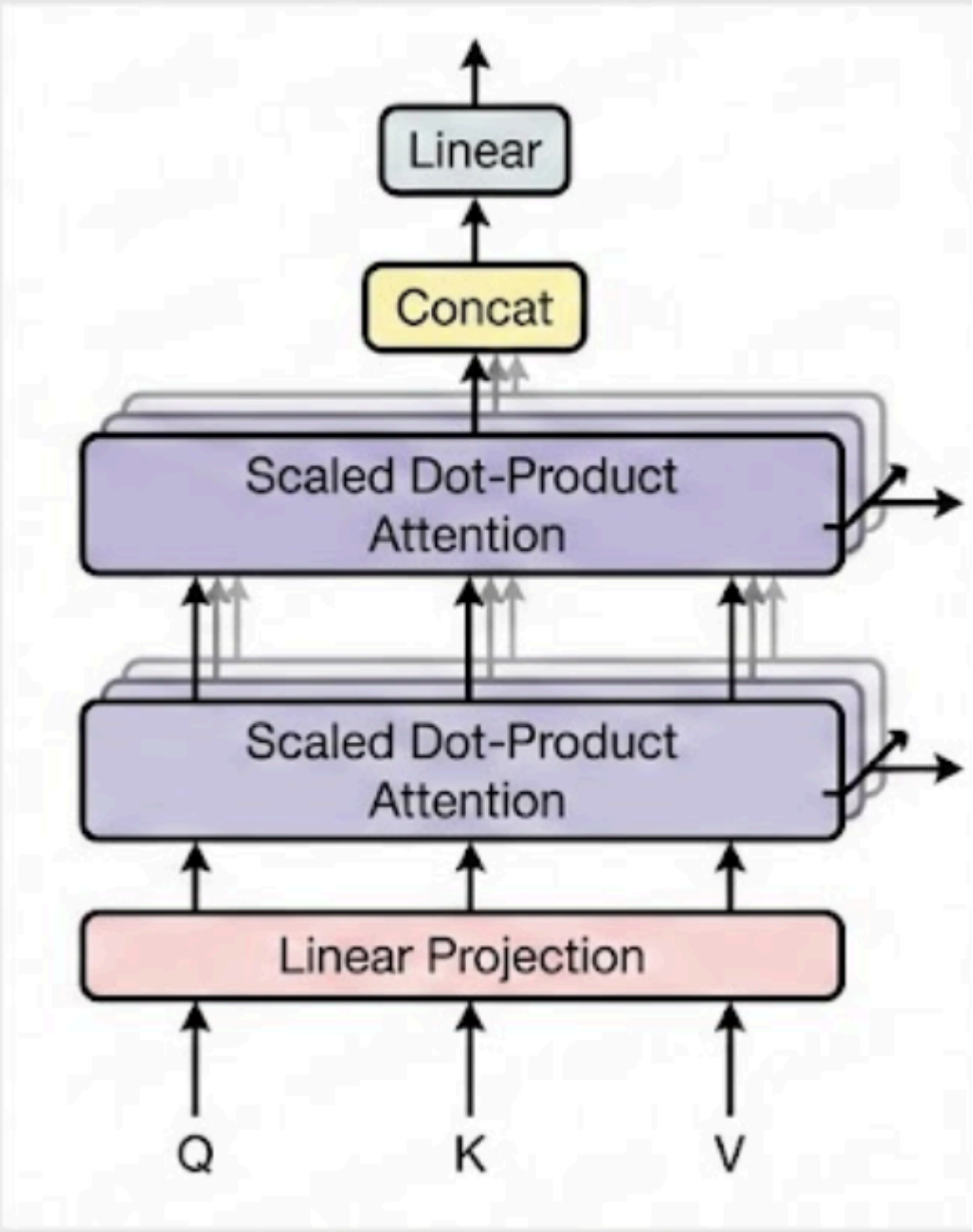
Transformer 架构与注意力机制



1. 缩放点积注意力 (Scaled Dot-Product Attention)

Transformer 的核心在于自注意力机制。给定查询矩阵 Q、键矩阵 K 和值矩阵 V，其计算公式为：

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

其中 $\sqrt{d_k}$ 是缩放因子，用于防止点积过大导致梯度消失。

2. 多头注意力 (Multi-Head Attention)

为了捕捉不同子空间的特征，我们将上述计算并行执行 h 次：

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O \tag{2}$$

3. 算法复杂度对比

层类型 (Layer Type)	复杂度 (每层)	顺序操作数	最大路径长度
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent (RNN)	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(\log_k(n))$

4. 架构实现 (Python/PyTorch)

以下是核心 Attention 层的简化代码实现：

```python
import torch
import torch.nn.functional as F

def scaled_dot_product_attention(query, key, value):
    d_k = query.size(-1)
    scores = torch.matmul(query, key.transpose(-2, -1)) / math.sqrt(d_k)
    p_attn = F.softmax(scores, dim=-1)
    return torch.matmul(p_attn, value), p_attn
```