Self-Attention Computation Process

1

Compute Query, Key, Value Matrices

Generate Q, K, V through linear projections

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

↓

2

Calculate Attention Scores

Compute dot product between queries and keys

$\mathrm{Scores} = QK^T$

↓

3

Scale by $\sqrt{d_k}$

Normalize scores to stabilize gradients

$\mathrm{Scaled} = QK^T / \sqrt{d_k}$

↓

4

Apply Softmax

Convert scores to attention weights (probabilities)

$\mathrm{Attention} = \mathrm{softmax}(QK^T / \sqrt{d_k})$

🎯

Higher scores indicate stronger relationships

🔗

Scores determine how much each token attends to others

💡 Numerical Example

Setup: 5 tokens (sequence length), d_{model} = 3 (hidden dimension), d_k = 3

① Input Matrix X (5×3):

1.0	0.5	0.2
0.8	1.2	0.3
0.6	0.9	1.1
1.1	0.4	0.7
0.9	0.7	0.8

Each row represents one token's embedding

Weight Matrices W_Q, W_K, W_V (3×3):

W_Q

1	0	0
0	1	0

W_K

1	0	0
0	1	0

W_V

1	0	0
0	1	0

0	0	1

0	0	1

0	0	1

Using identity matrices for simplicity → Q = K = V = X

② Attention Scores = QK^T (5×5):

1.29	1.46	1.48	1.35	1.41
1.46	2.17	1.89	1.61	1.69
1.48	1.89	2.38	1.69	1.91
1.35	1.61	1.69	1.66	1.63
1.41	1.69	1.91	1.63	1.94

Higher values indicate stronger similarity between token pairs

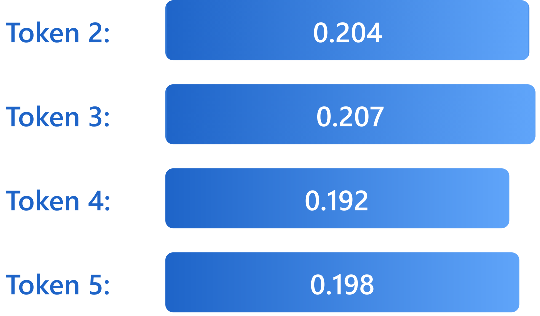
③ Scaled Scores = $QK^T / \sqrt{d_k}$ (÷√3 ≈ 1.732):

0.74	0.84	0.85	0.78	0.81
0.84	1.25	1.09	0.93	0.98
0.85	1.09	1.37	0.98	1.10
0.78	0.93	0.98	0.96	0.94
0.81	0.98	1.10	0.94	1.12

Scaling prevents gradients from becoming too small during backpropagation

④ Attention Weights after Softmax (First Row Example):

Token 1: 0.182



✓ Sum = 1.0 (normalized probability distribution)

✓ Token 1 attends most to Token 3 (0.207) and least to itself (0.182)