Transformer: Calculate One Attention Step

Name: _____ Date: _____

Attention mixes value vectors using weights made from a query and keys. These are hand-made numbers for one head, not learned word meanings or a complete Transformer.

One example, already done

q=[1,0]; keys=[1,0],[0,1],[1,1]. Dot products are 1,0,1. Divide by sqrt(2), then softmax: exp(score)/sum(exp(scores)). Weights are about 0.401,0.198,0.401.

Your turn

1. Check the weights sum to approximately 1. Values are [1,0],[0,1],[2,1]. Compute BOTH dimensions of the weighted mix, one at a time.

2. Mask position 3 so a query at position 2 cannot read the future. Its weight is 0. Re-normalize positions 1 and 2; the weights are about 0.670,0.330. Compute the new output.

3. Change q to [0,1], without masking. Which dot products change? Are attention weights the same thing as the probabilities of the next word?

Transformer: Calculate One Attention Step

Make it visible

Change one thing

If all keys are identical and nothing is masked, what are the three weights? What positional information would a real sequence model still need?

My puzzle - make it in class

Choose three small value vectors. Ask a parent to mix them with weights 0.5,0.25,0.25. Keep both output dimensions in your answer key.

At home (with a parent)

Give a parent three computed weights, check they sum to one and calculate the mixed value.

My parent tried my puzzle: Yes / Not yet Parent initials: _____