Attention

Excerpts from Chapter 10 of
Speech and Language Processing,
Jurafsky and Martin, Feb. 3, 2024 draft
Worked example developed with Claude Sonnet 5.5
Michael Wollowski

1

Reminder: Dot Product

- Definition:

$$\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^{n} a_i b_i$$

- Outcome: Can be used to calculate the similarity between two vectors.

2

Bird's Eye View of Attention in Transformers

- Starting at the bottom, we calculate the dot product between:
  - X_3 and X_3
  - X_3 and X_2
  - X_3 and X_1
- This should give us a similarity of X_3 to the prior tokens, X_1, X_2 and itself.

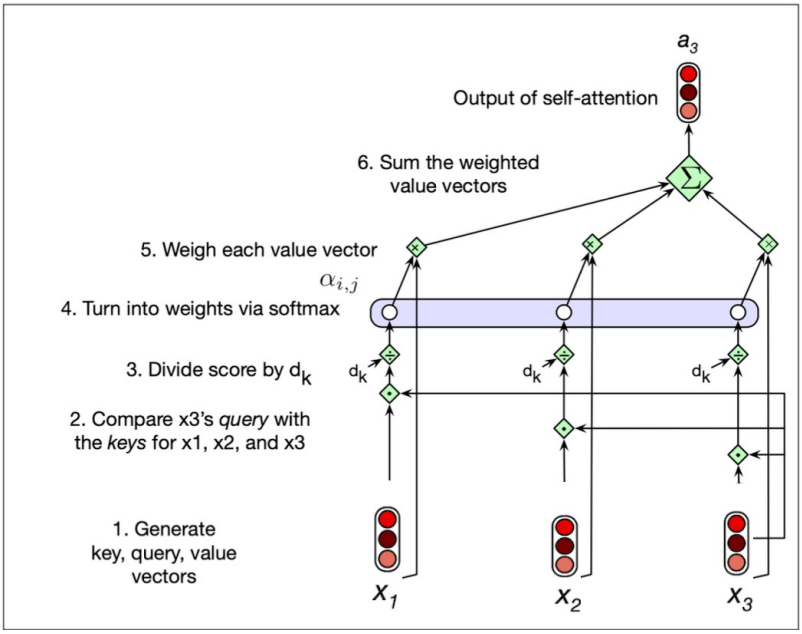


Image source: Modified from Speech and Language Processing, Jurafsky and Martin, Feb. 3, 2024 draft

3

Bird's Eye View of Attention in Transformers

- Next, we scale the values of the dot product.
- Otherwise, the values may be quite large and impede training, due to loss of gradients.
- We divide by the square root of the dimensionality of the key vector, d_k

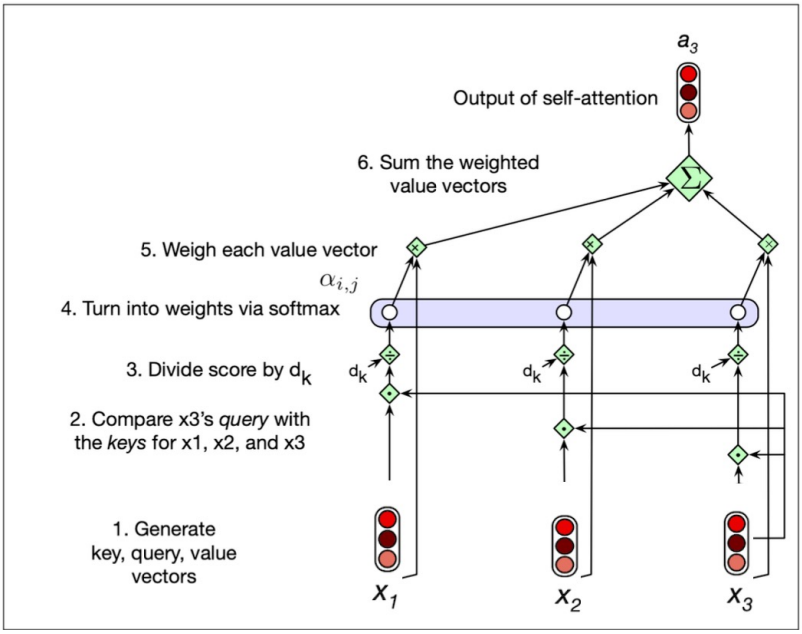


Image source: Modified from Speech and Language Processing, Jurafsky and Martin, Feb. 3, 2024 draft

4

Bird's Eye View of Attention in Transformers

- Now, we run the values through softmax to obtain weights.
- These weights are now used to determine the relevance of each of X_1 to X_3
- Finally, we produce the weighted sum with the X_1, X_2, and X_3.

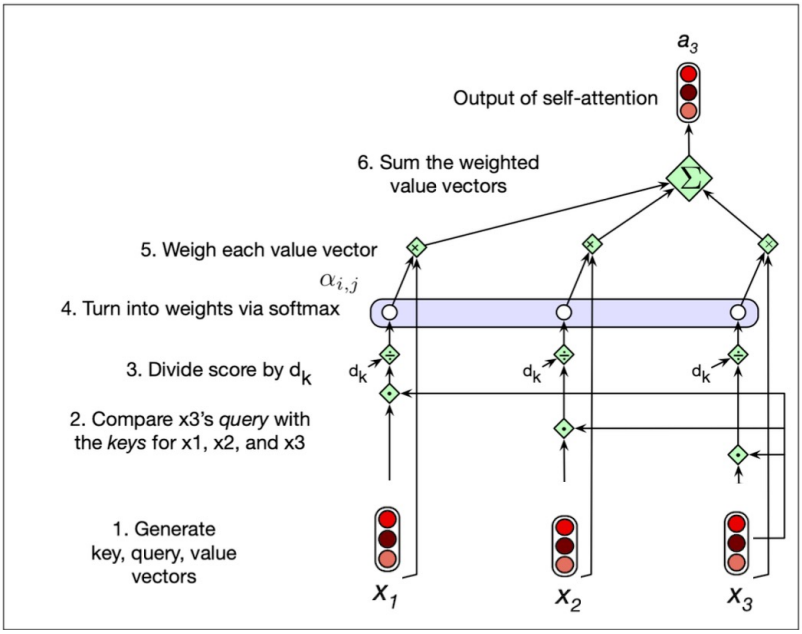


Image source: Modified from Speech and Language Processing, Jurafsky and Martin, Feb. 3, 2024 draft

5

Reminder: Softmax

- Definition:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- The softmax function takes as input a vector and:
  - turns each component into an interval (0,1)
  - the components will add up to 1,
  - they can be interpreted as probabilities

6

Transformers – for reals

- We should add some weights.
- After all that is what NNs are all about.
- We want to have a trainable similarity measure, rather than a static one.

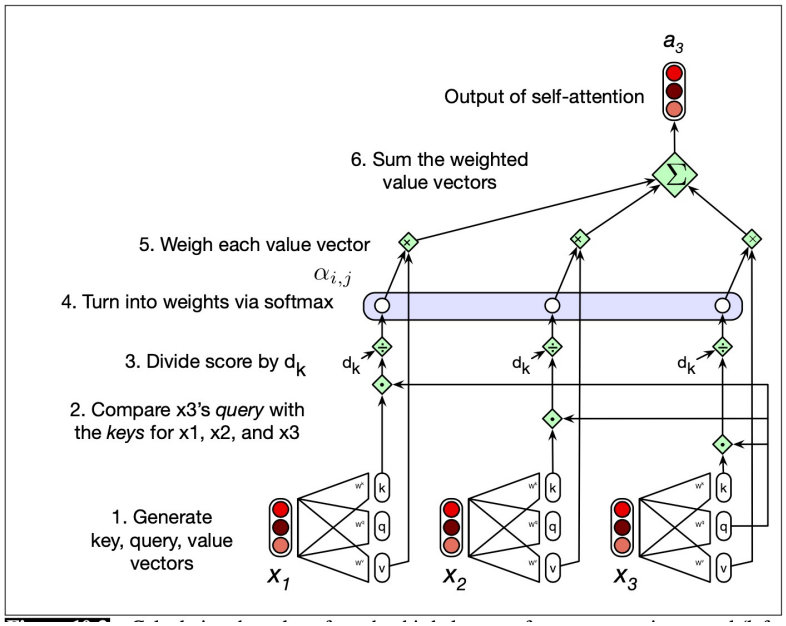


Image source: Speech and Language Processing, Jurafsky and Martin, Feb. 3, 2024 draft

7

Transformers – for reals

- Each X_i is multiplied by three different weight matrices, W^k, W^q and W^v
- They are the same three for every token, to produce a:
  - key **k**,
  - query **q**
  - value **v**.

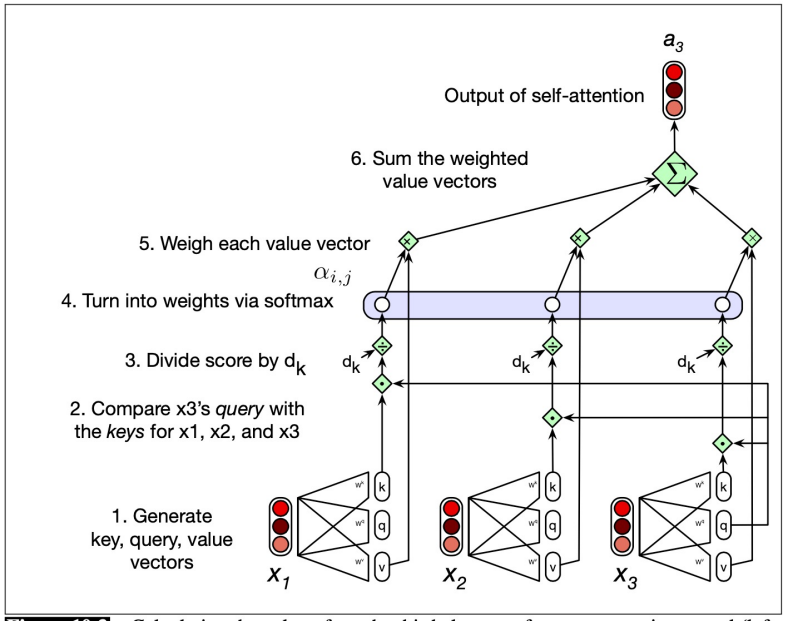


Image source: Speech and Language Processing, Jurafsky and Martin, Feb. 3, 2024 draft

8

Keys, Queries and Values

- Consider the phrase: “The cat sat <end>”
- Suppose we process the token “sat”
- What do we wish to learn about the “sat”?
- Who is sitting?
- In this case, the cat.

9

Keys, Queries and Values

- Continuing with the phrase: “The cat sat <end>“
- To learn more about the token “sat”, we issue a **query** from “sat”:
  - “who is doing the sitting?”
- Every token offers a **key**, which describes what that token has to offer.
- The query is compared with each key by a dot product.
- The key of “cat” should match the query well.
- We score all the keys the query can see (“The”, “cat” and “sat”).
- Every token also carries a **value**, the information it contributes.
- The output for “sat” is the sum of the values, weighted by those weights, so “cat” can contribute strongly.

10

Transformers – for reals

- In the diagram, the attention is determined for X_3.
- The query of X_3 is multiplied (dot product) with the key of X_1, X_2 and X_3.
- The scores are scaled and passed through softmax.
- This gives one weight for each of the three tokens.
- Output a_3 is the sum of the values of X_1, X_2 and X_3, each multiplied by its softmax weight.

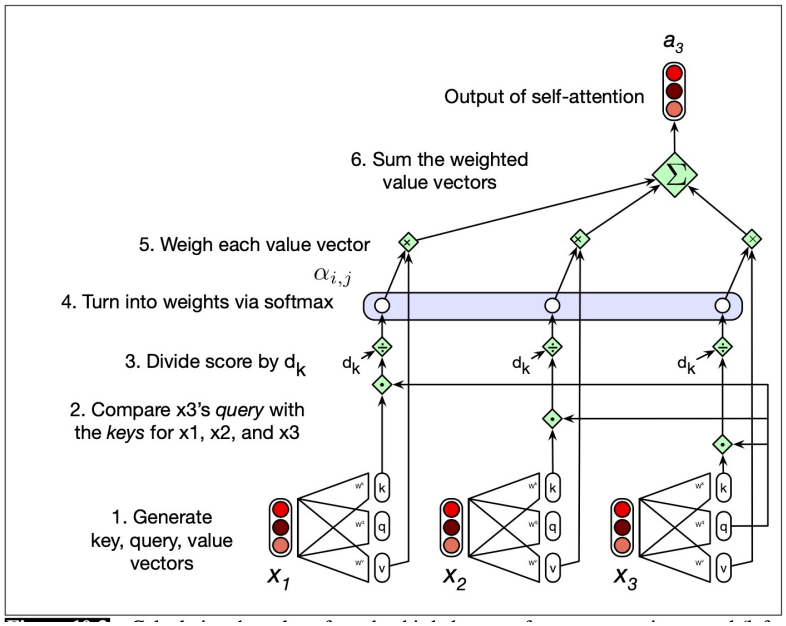


Image source: Speech and Language Processing, Jurafsky and Martin, Feb. 3, 2024 draft

11

Multi-head Attention

- In the context of:

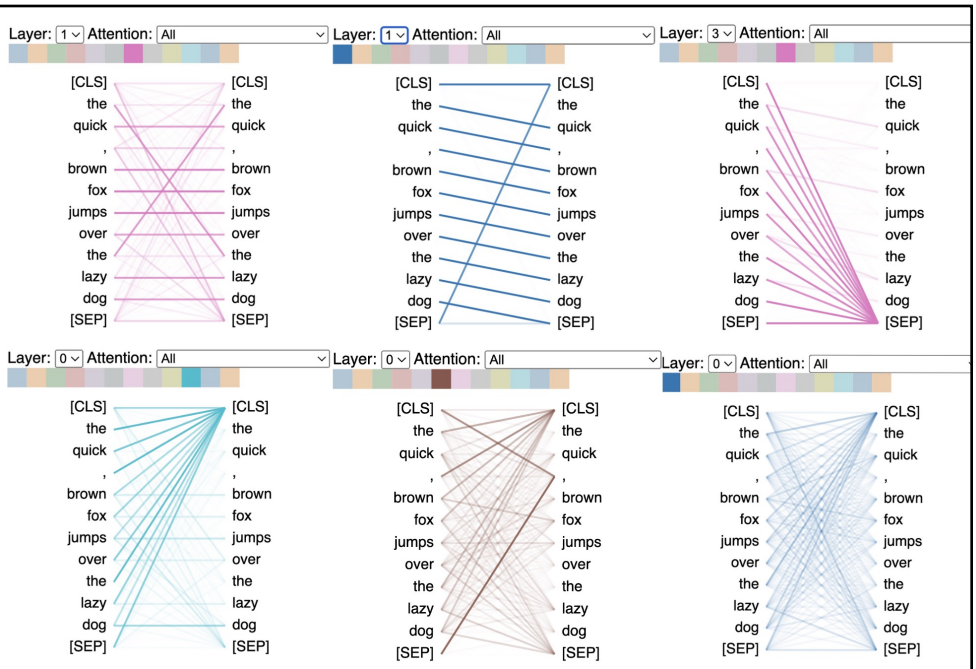


- We will add more than one attention head!
- More attention heads, more weights, more things to pay attention to.
- Recall that for CNNs, we applied several filters to a matrix, to "look" at different aspects of an image.

Image source: https://muppet.fandom.com/wiki/Me_Want_Cookie?file=MeWantCookie.jpg

12

BertViz shows that attention captures various patterns in language, including positional patterns, delimiter patterns, and bag-of-words.

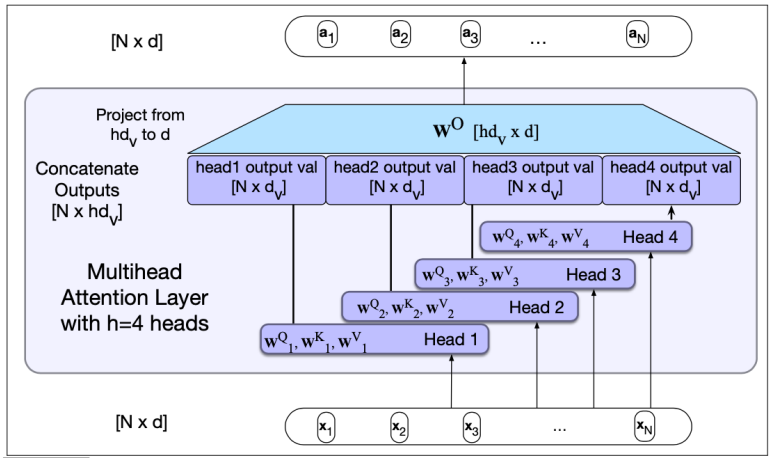


Source: https://www.comet.com/site/blog/explainable-ai-for-transformers/

13

Multi-head Attention

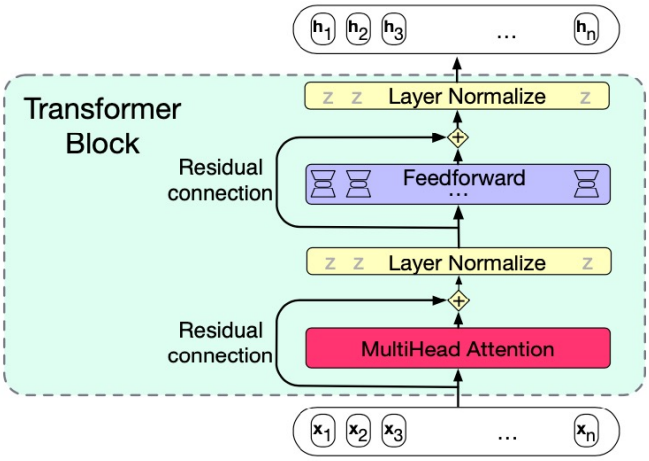
- Each of the multi-head self-attention layers is provided with its own set of key, query and value weight matrices.
- The outputs from each of the layers are concatenated.
- They are then projected to *d*.
- Thus producing an output of the same size as the input.



14

Transformer Blocks

- The self-attention calculation lies at the core of what is called a **transformer block**.
- In addition to the self-attention layer, it includes feedforward layers, residual connections, and normalizing layers.



15