self → output

[Attn of each word relative to other word].

- So,
  - Q, K, V init W_Q, W_K, W_V
  - → ~~Attn~~ ~~score~~ $(Q \times K)$
  - → scale
  - → softmax
  - → weighted sum (softmax × V)
  - → Attn score for token.
    - ↳ self Attn.

(*) Attn Head Multi°

Cont.... to next Note.

More about word embeddings

- Other ~~embedding~~ Numeric Reps of word or sequence does not capture the semantic meaning of the word.

(*) Word Embedding [captures Semantic context].

- Word embedding uses high dimensional vectors to represent each word.
- For example, if we train Word2Vec model using [CBow, Skip Gram] we'll reps each word in a very high dimensional vectors.

✓ Understand with example.

→ Word embeddings for similar meaning are closer and far for dissimilar words.

[Average Meaning Problem]

e.g we've the below dataset that we'll be using to create [our word embedding] or dictionary.

S_1 => An apple a day keeps a doctor away
S_2 => Apple is healthy -> Taste
S_3 => Apple is better than orange -> Taste
S_4 => Apple makes great phones -> Tech

Say, we're in a stage of training where we're about to generate embedding for the word Apple [Num exp]

- We're exp each word in 2D i.e [taste, Technology]

(A) Note

- the problem with word Embedding is that it captures the [Average Meaning]. So, it describes, on an average any word is used with what meaning of the training data.

So, in our example, if we've huge dataset where, ~~about~~ Apple (word) is used mostly with the meaning of Taste, then the [Test] element will have higher value.

On the other side, if we flip the case where our dataset has Apple used with meaning of Technology, then the [Technology vector (element) will have higher value.

so, word embedding for each word depends on the dataset's meaning that is used or speaks on average. [Prevent Biased Datasets]

eg Apple(Fruit, Taste) => [0.9, 0.2]
Apple[Tech] => [0.1, 0.8]

(*) So, how is this a problem?

- Because [word embeddings] are created only once, and used time and again. i.e) [Static]

For example: After training a embedding model, let's say the embedding of apple is [[0.9 0.3]]
[Taste, Tech]

Now, if we've a sentence Apple launched a new phone while I was eating an orange.

Here, Apple would refer to [Fruit] because of the training.

But what we actually needed was [Contextual Embedding.] These [0.9 0.3] value should have changed dynamically

As in the sentence these are words like launched, ~~model~~, phone these words should've increased the Technology dimension and lowes the Taste. It should not be confused by the word Orange.

=> to solve this problem we've Self Attention.

Self Attention generates Dynamic Embedding i.e Contextual Embedding from the word embedding.