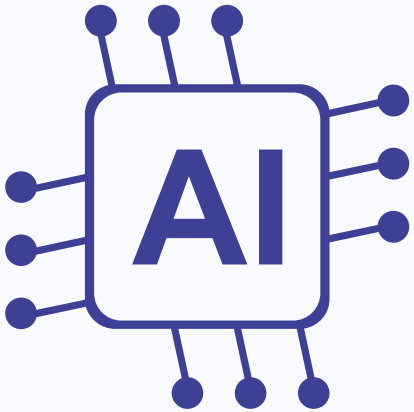
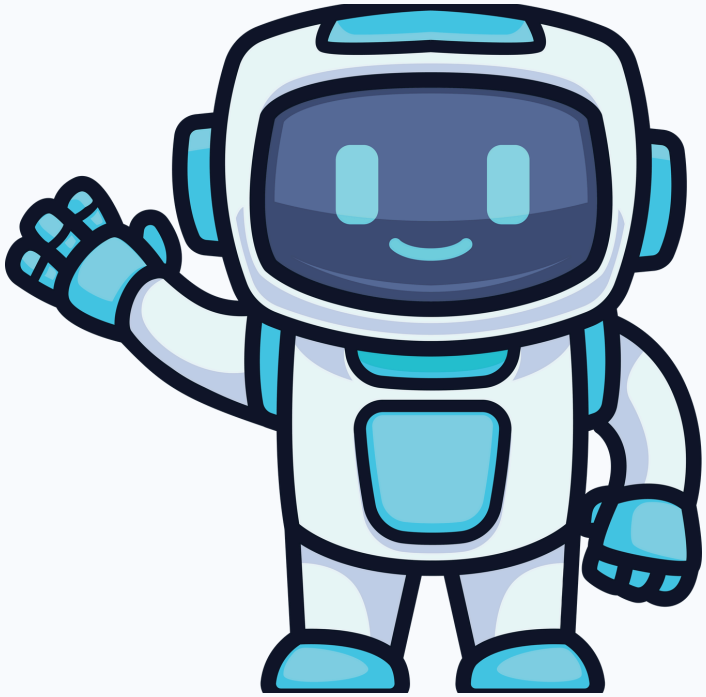
Transformer Chatbot

LiveAi

Unlocking
Your Potential,
Unleashing
Your Success





DataSet Identification

Cornell Movie-Dialogs Corpus

LiveAi

Each line in the movie_lines.txt file of the dataset follows this structure:

lineID +++$+++ characterID +++$+++ movieID +++$+++ character name +++$+++ text of the utterance

L1044 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ They do to!

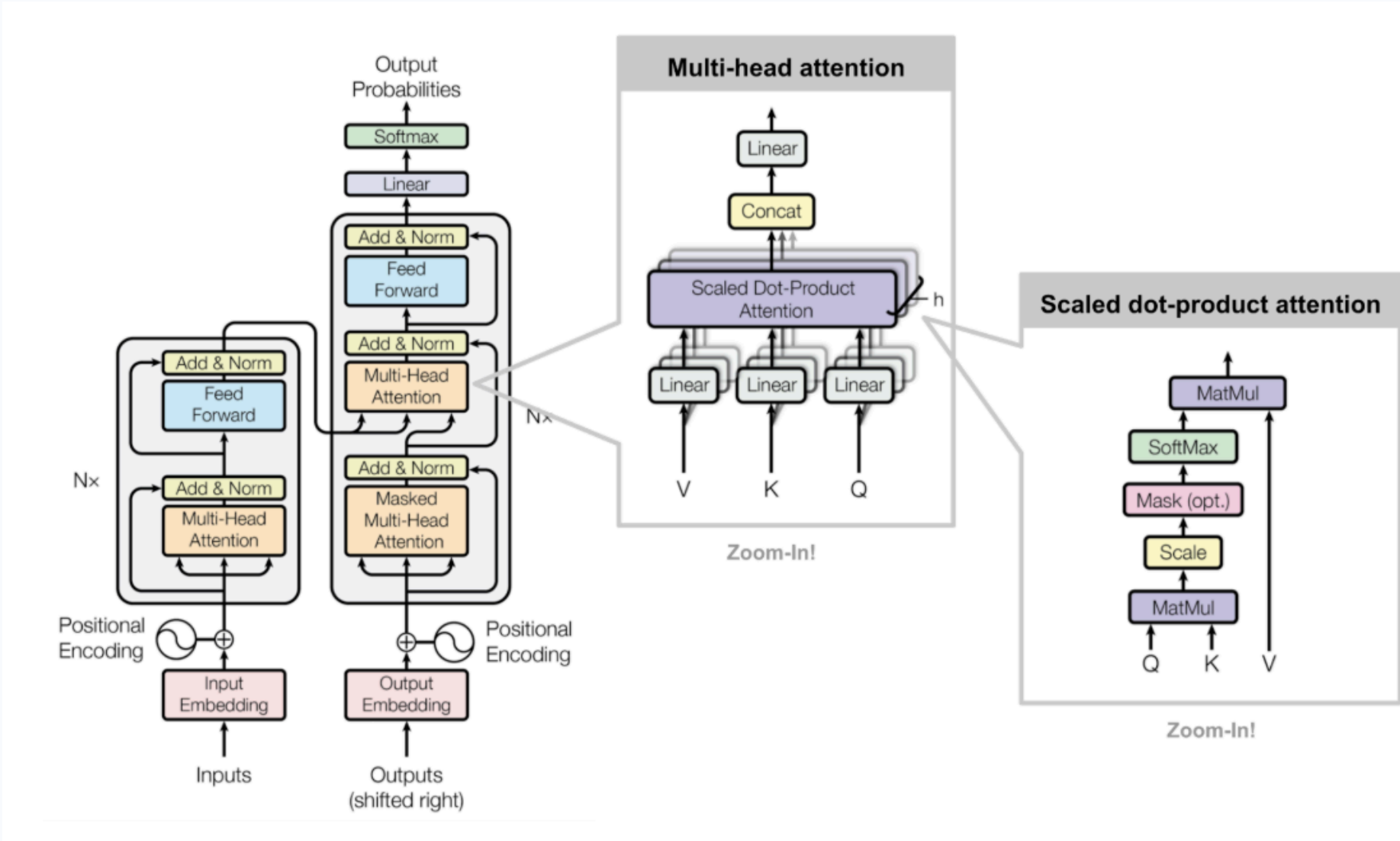
L1045 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ They do not!

movie_conversations.txt links line IDs from movie_lines.txt to reconstruct full conversations

u0 +++$+++ u2 +++$+++ m0 +++$+++ ['L194', 'L195', 'L196', 'L197']

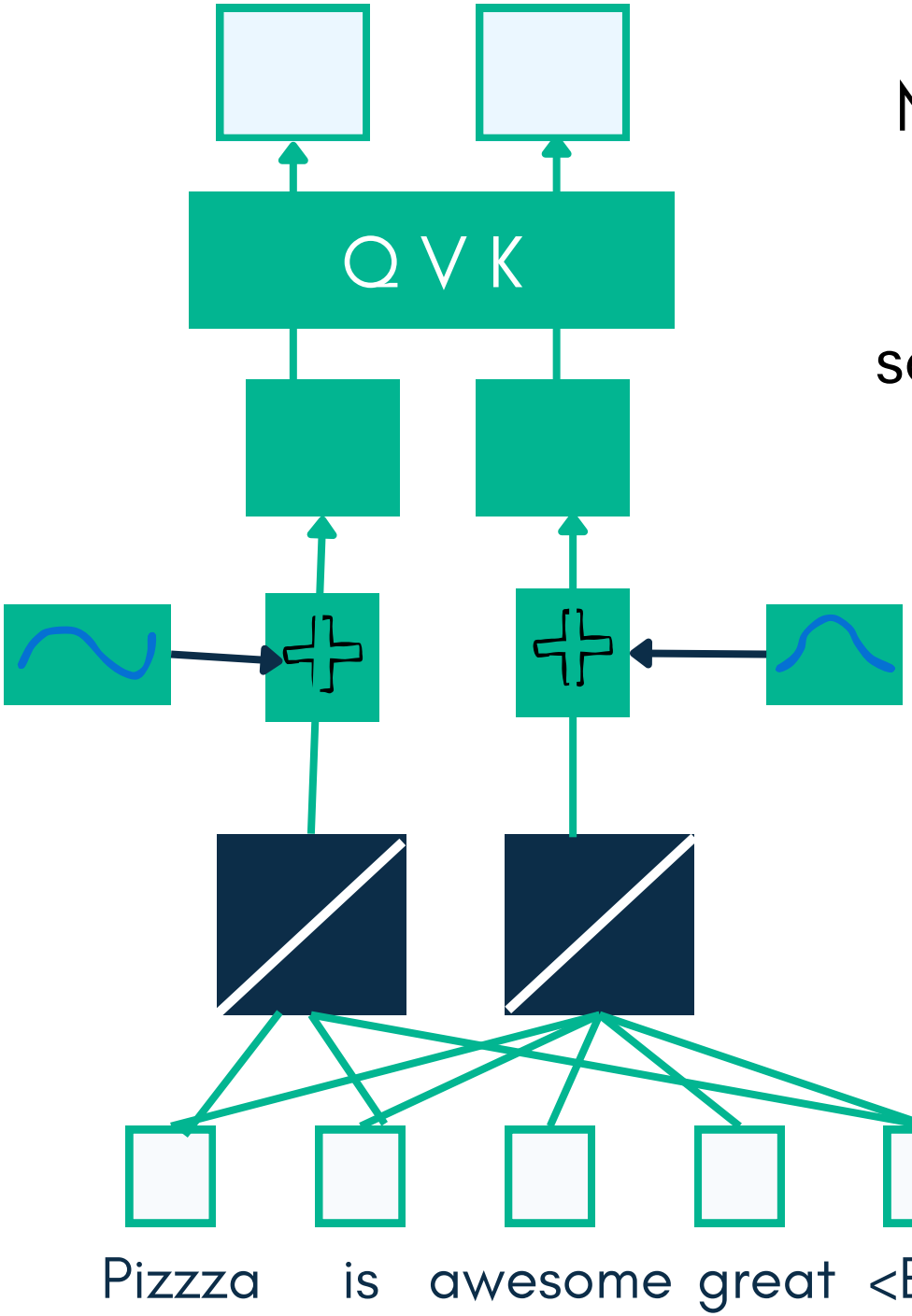
L194 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Can we make this quick? Roxanne Korrine and Andrew Barrett are having an incredibly horrendous public break-up on the quad. Again.
L195 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Well, I thought we'd start with pronunciation, if that's okay with you.
L196 +++$+++ u0 +++$+++ m0 +++$+++ BIANCA +++$+++ Not the hacking and gagging and spitting part. Please.
L197 +++$+++ u2 +++$+++ m0 +++$+++ CAMERON +++$+++ Okay... then how 'bout we try out some French cuisine. Saturday? Night?

Transformer Architecture



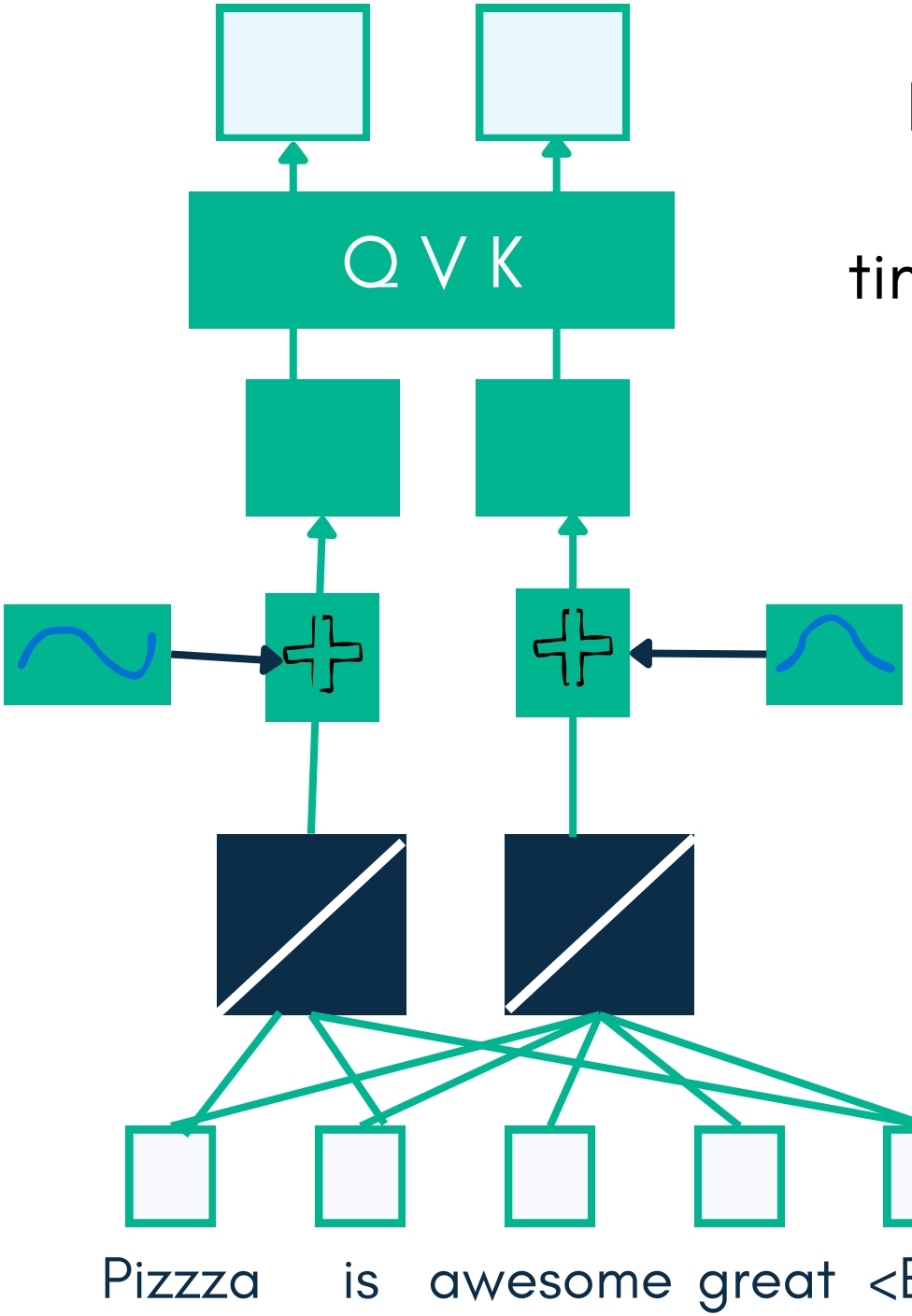
Transformers use **word-embeddings** to convert words into numbers...
Positional encdoding to keep track of words ....
Transformers have **self-attention** that associate word similiarity within input and output sequence....
Encoder-decoder attention to keep track of things between the input and the output phrases.... and are not lost in translation .....

Our example includes a chat conversation
Actor 1: Pizza is awesome great <EOS> --- encoder
Actor 2. That's really good<EOS> - decoder

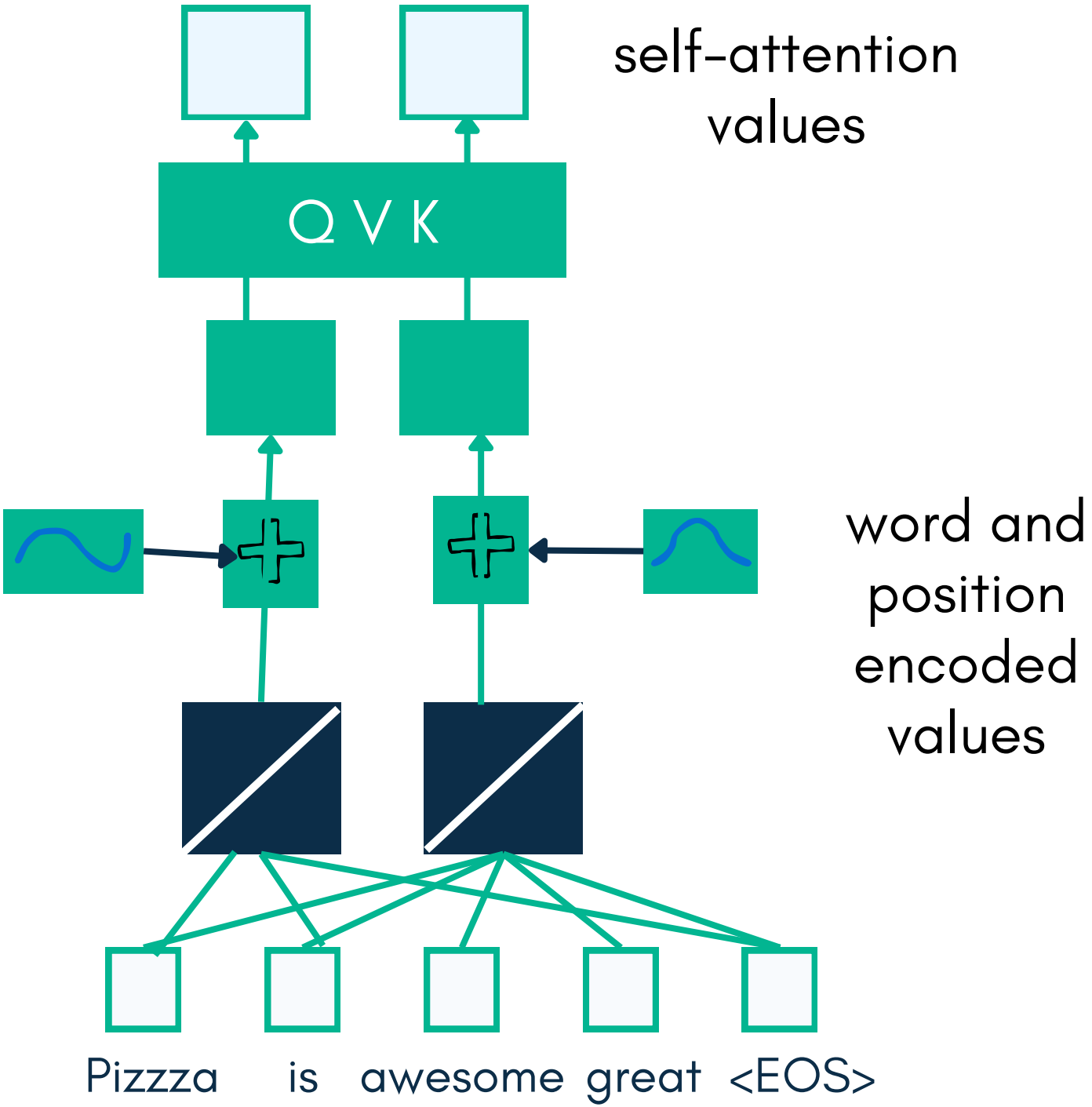


No matter how many words are input into the Transformers we reuse the same set of weights for self-attention queries,keys and values

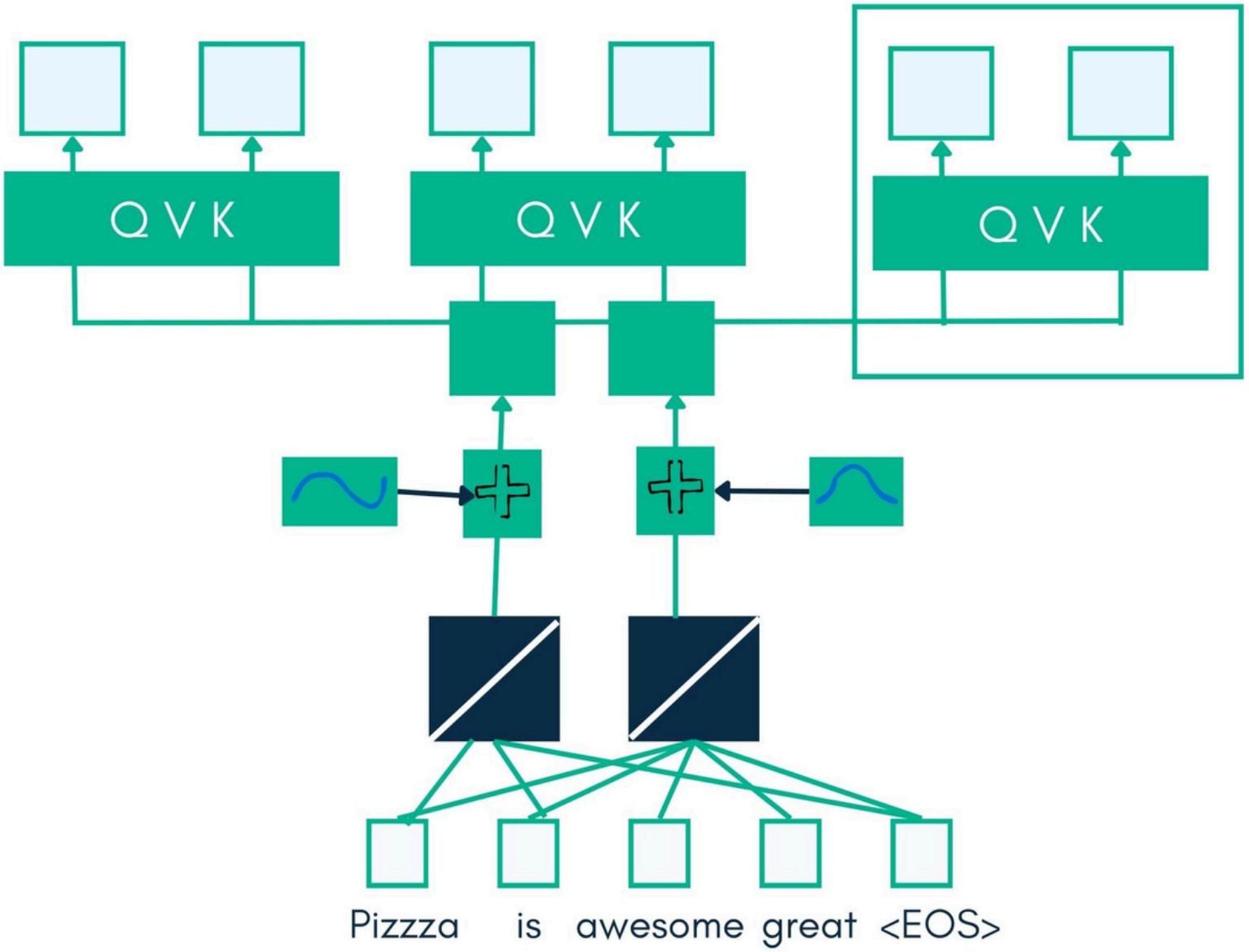
Self-attention queries,keys and values for all the words are calculated simultaneously

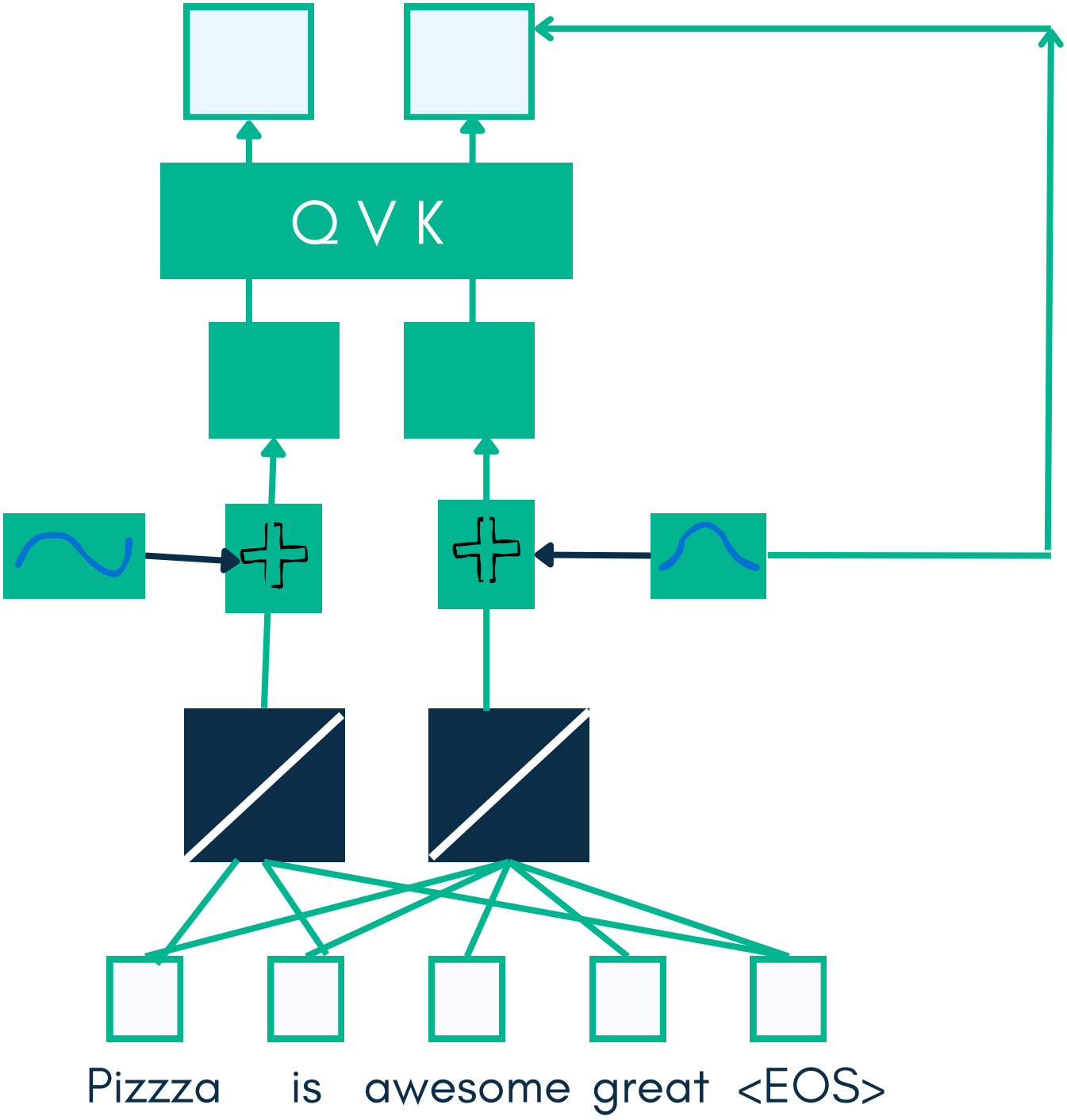


Because we can do all the computations at the same time - **Transformers** can take advantage of parallel computation


self-attention values
Q V K
word and position encoded values
Pizzza
is
awesome
great
<EOS>

Finally, if you have a number of sellf-attention units together you will get multi-head attentions

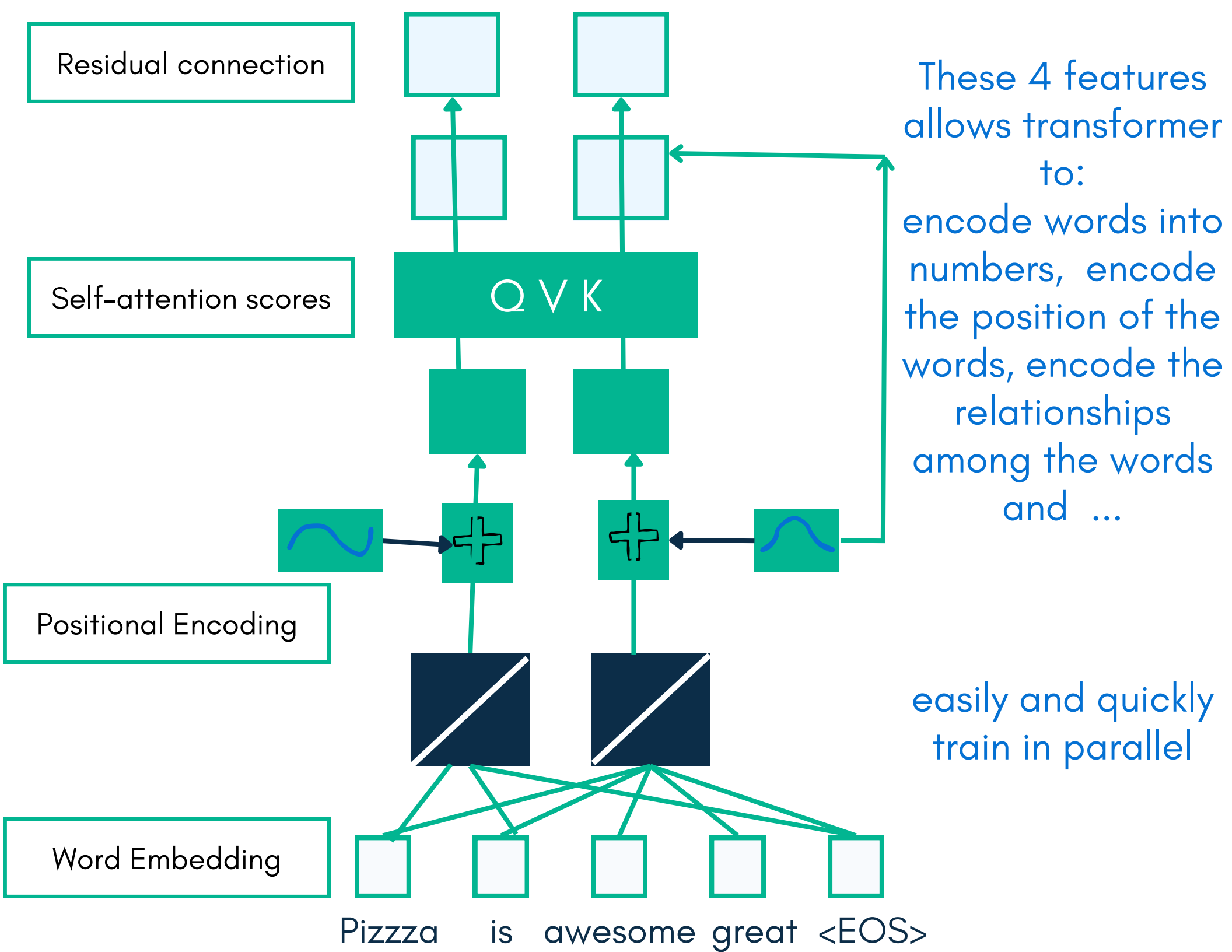


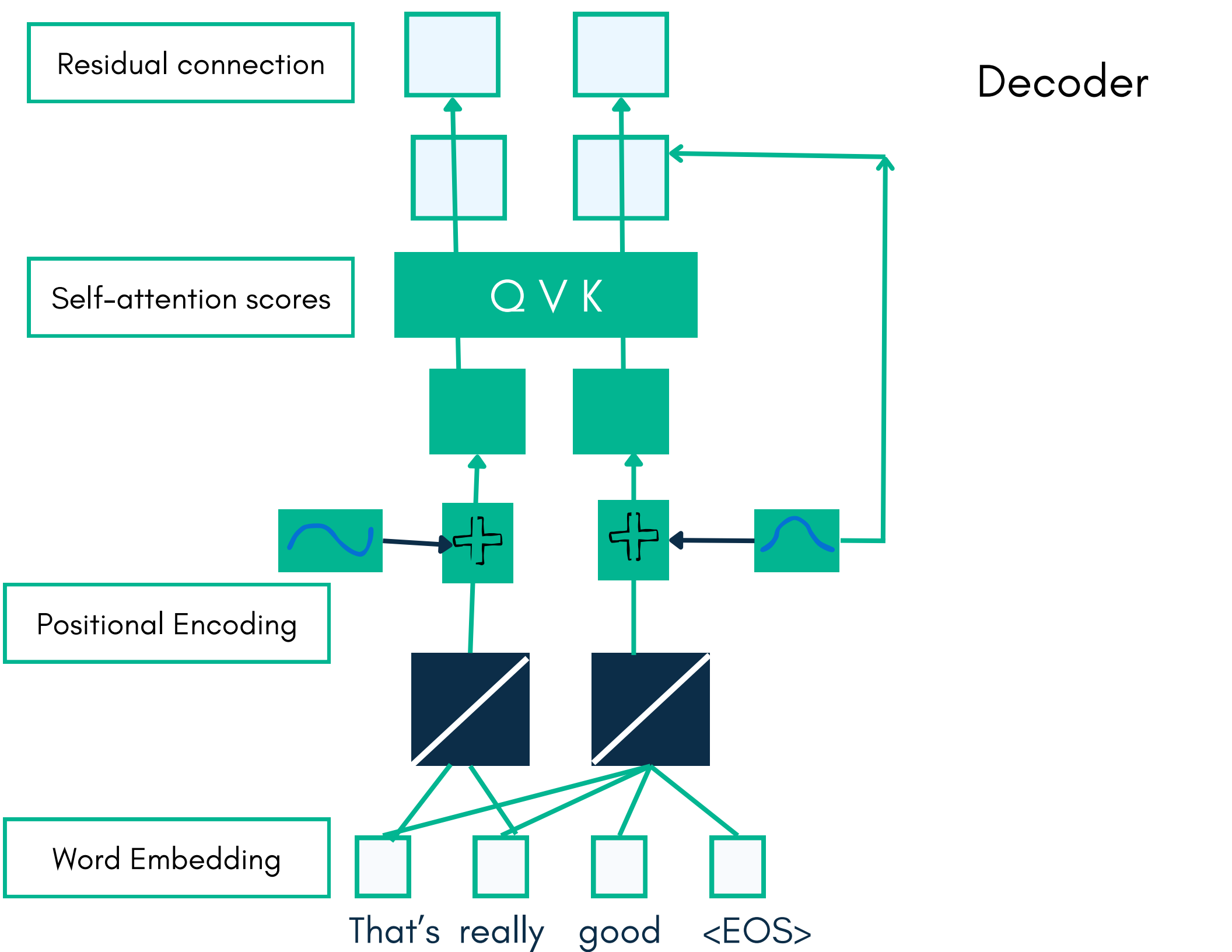


We take the position encoded values and them to the self-attention values ..

These by-passes are called Residual Connections and they make it easier to train complex neural networks.....

It allows the self-attention layer to establish relation among the words without preserving word embedding and postional encoding information.


Residual connection
Self-attention scores
Q V K
Positional Encoding
Word Embedding
Pizzza
is
awesome
great
<EOS>
These 4 features allows transformer to:
encode words into numbers, encode the position of the words, encode the relationships among the words and ...
easily and quickly train in parallel


Residual connection
Self-attention scores
Q V K
Positional Encoding
Word Embedding
That's really good <EOS>
Decoder

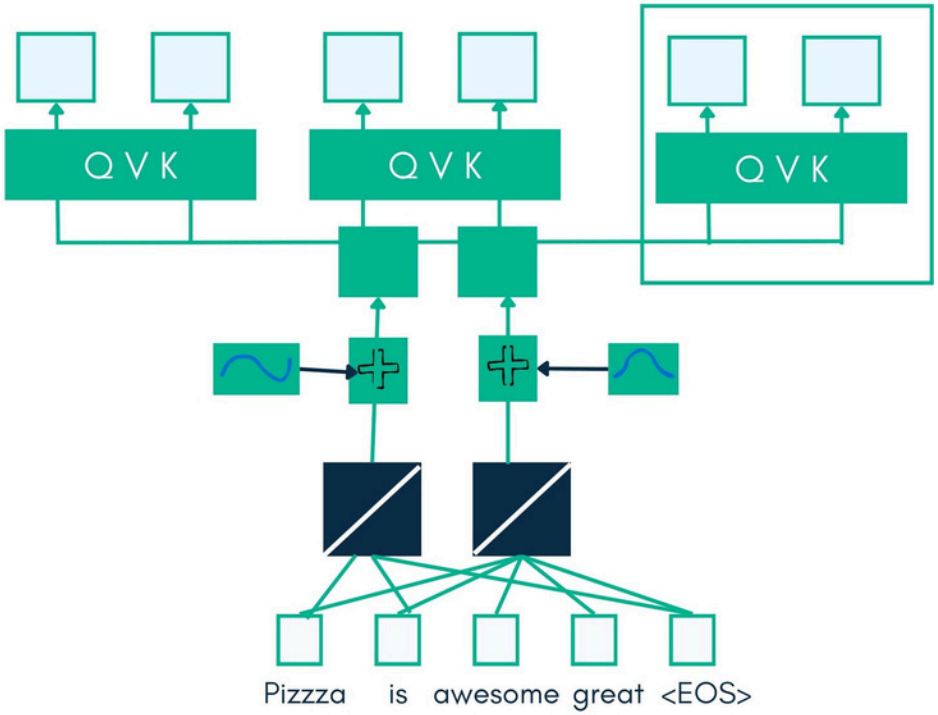
So far we talked about how self-attention keeps track of how words are related with a sentence ... However, since we are creating a conversational chatbot .... we need to keep track between the input sentence and the output sentence.

Its important for the **Decoder** to keep track of the significant words in the input.

So, the main idea of **encoder-decoder attention** is to keep track of the significant words in the input.

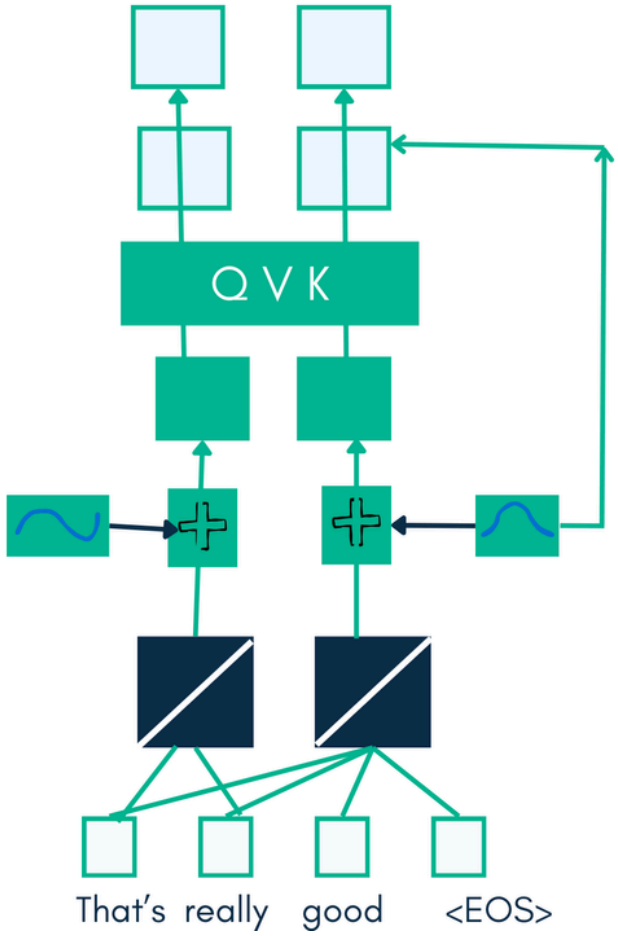
we create 2 new values to represent the Query of the <EOS> token in the Decoder

Q: [-0.9, 2.6]



Then we create keys for each word in the Encoder

K: [-4.5, 2.1, -0.5, 4.1]



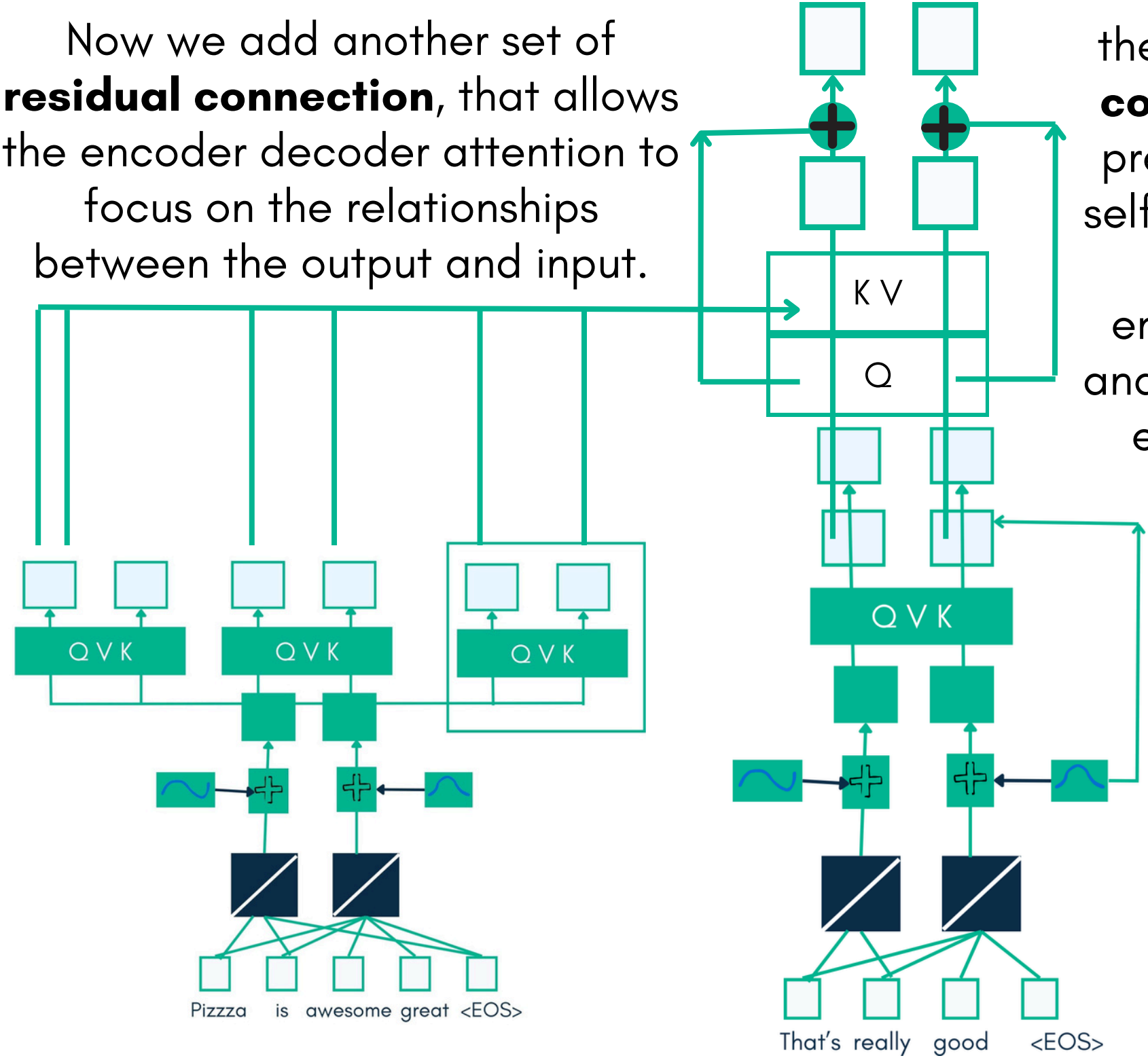
We calculate the similarities between <EOS> token in the decoder and each word in the encoder ...

By calculating the dot products.

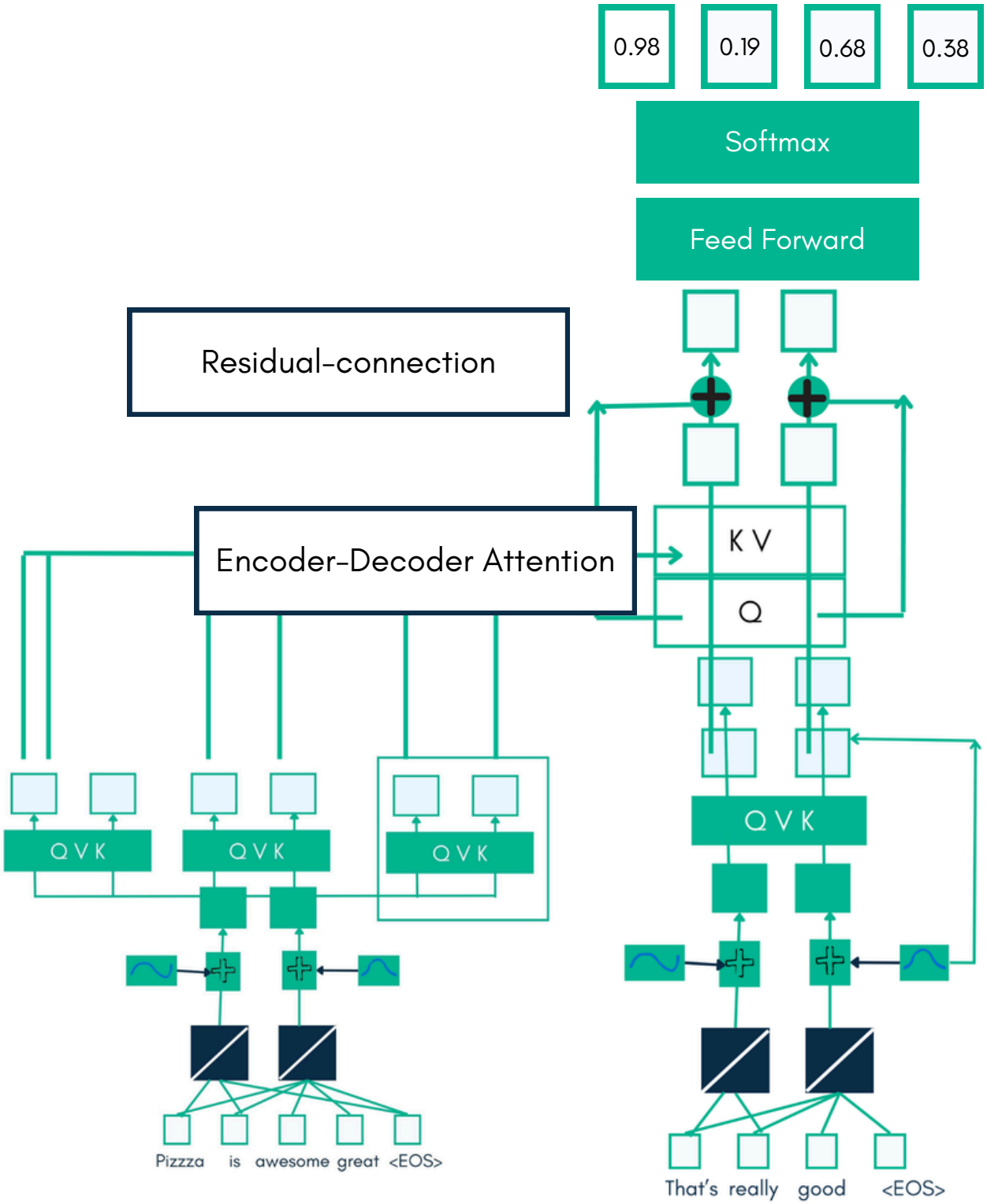
We then run the similarities through a softmax function.

We then calculate values of each input word and scale those values by softmax percentages. Finally, add the pair of scaled values together to get Encoder-Decoder attention values.

Now we add another set of **residual connection**, that allows the encoder decoder attention to focus on the relationships between the output and input.



the **residual connection**, preserve the self-attention, word embedding and positional encoding



Finally, we need to connect a fully connected layer to calculate the probabilities of the tokens "That's", "really", "good", "<EOS>". The fully connected layer has one input for each value of the current token, so in this case **2** ...

and one output for each token in the output vocabulary wihich is **4** in this case

We run the final Softmax function to select the sentence "That's really good <EOS>"