Attention and Transformers

1. Text tokenization

- The first step in processing text is to cut it into pieces, called tokens.
- There are many variations of how to do it, and we won't go into details, for example BERT uses WordPiece tokenization. This means that tokens correspond roughly to words and punctuation, although a word can also be split into several tokens if it contains a common prefix or suffix.
- Words can even be spelled out if they have never been seen before.

2. Text embedding

- The second step is to associate each token with an embedding, which is a vector of real numbers.
- There are many ways to create embedding vectors. Fortunately, already trained embeddings are often provided by research groups, and we can just use an existing dictionary to convert the WordPiece tokens into embedding vectors.
- The embedding of tokens into vectors is an achievement in itself: The values inside an embedding carry information about the meaning of the token, but they are also arranged in such a way that one can perform mathematical operations on them, which correspond to semantic changes, like changing the gender of a noun, or the tense of a verb, or even the homeland of a city.
- Embeddings are associated with tokens by a straight dictionary lookup, which means that the same token always gets the same embedding, regardless of its context.
- This is where the attention mechanism comes in, and specifically for BERT.

3. Context and attention

The scaled dot-product self-attention

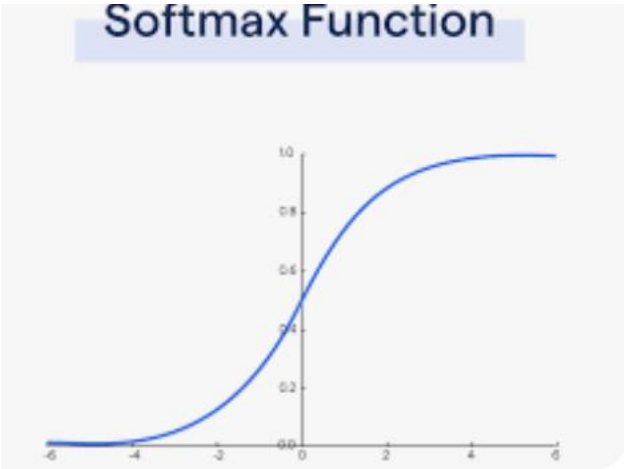
Attention transforms the default embeddings by analyzing the whole sequence of tokens, so that the values are more representative of the token they represent in the context of the sentence.

4. Self-attention mechanism

- Each token is initially replaced by its default embedding, which in this case is a vector with 768 components.

- We start by calculating the scalar product between pairs of embeddings, we have the first embedding with itself. When the two vectors are more correlated, or aligned, meaning that they are generally more similar, the scalar product is higher, and we consider that they have a strong relationship. If they had less similar content, the scalar product would be lower and we would consider that they don't relate to each other.

- We go on and calculate the scalar product for every possible pair of embedding vectors in the input sequence. The values obtained are usually scaled down to avoid getting large values, which improves the numerical behavior. That's done by dividing by the square root of 768, which is the size of the vectors.



- Then comes the only non-linear operation in the attention mechanism, the scaled values are passed through a softmax activation function, by groups corresponding to each input token. So in this illustration, we apply the softmax column by column. What the softmax does is to exponentially amplify large values, while crushing low and negative values towards zero. It also does normalization, so that each column sums up to 1.

- Finally, we create a new embedding vector for each token by linear combination of the input embeddings, in proportions given by the softmax results. We can

say that the new embedding vectors are contextualized, since they contain a fraction of every input embedding for this particular sequence of tokens.

- In particular, if a token has a strong relationship with another one, a large **fraction** of its new contextualized embedding will be made of the related embedding. If a token doesn't relate much to any other, as measured by the scalar product between their input embeddings, its contextualized embedding will be nearly identical to the input embedding.

- For instance, one can imagine that the vector space has a direction that corresponds to the idea of nature. The input embeddings of the tokens RIVER and BANK should both have large values in that direction, so that they are more similar and have a strong relationship. As a result, the new contextualized embeddings of the RIVER and BANK tokens would combine both input embeddings in roughly equal parts. On the other hand, the preposition BY sounds quite neutral, so that its embedding should have a weak relationship with every other one, and little modification of the embedding vector would occur.

So there we have the mechanism that lets the scaled dot-product attention utilize context.

- First, it determines how much the input embedding vectors relate to each other using the scalar product.

- The results are then scaled down, and the softmax activation function is applied, which normalizes these results in a non-linear way.

- New contextualized embeddings are finally created for every token by linear combination of all the input embeddings, using the softmax proportions as coefficients.

5. Key, Query, and Value projections

- We don't have to use the input embedding vectors as is. We can first project them using linear projections to create the so-called Key, Query, and Value vectors.

- Typically, the projections are also mapping the input embeddings onto a space of lower dimension. In the case of BERT, the Key, Query, and Value vectors all have 64 components.

- Each projection can be thought of as focusing on different directions of the vector space, which would represent different semantic aspects. One can imagine that a Key is the projection of an embedding onto the direction of prepositions, and a Query is the projection of an embedding along the direction of locations. In this case, the Key of the token BY should have a strong relationship with every other Query, since BY should have strong components in the direction of prepositions, and every other token should have strong components in the direction of locations.

- The Values can come from yet another projection that is relevant, for example the direction of physical places.It's these values that are combined to create the contextualized embeddings.In practice, the meaning of each projection may not be so clear, and the model is free to learn whatever projections allow it to solve language tasks the most efficiently

6. Multi-head attention

In addition, the same process can be repeated many times with different Key, Query, and Value projections, forming what is called multi-head attention.

Each head can focus on different projections of the input embeddings. For instance, one head could calculate the preposition/location relationships, while another head could calculate subject/verb relationships, simply by using different projections to create the Key, Query, and Value vectors.

The outputs from each head are concatenated back in a large vector. BERT uses 12 such heads, which means that the final output contains one 768 component contextualized embedding vector per token, equally long with the input.