**What is Transformer architecture?**

Transformer was first introduced in the paper Attention Is All You Need. It was/is a novel architecture which was able to handle long range dependencies in **sequence-to-sequence** modelling solely based on attention mechanism without using convolution or recurrence. This helped the architecture be more parallelizable and taking less time to train compared to then state of the art of models such as recurrent neural network, long short-term memory, and gated recurrent neural networks. It achieved state of the art results on translation datasets.

The Transformer architecture consists of encoder and decoder block. These blocks are multiple identical decoders and encoder stacked on top of each other. The number of units in the blocks are same and is a hyperparameter, which in the paper was 6.

Diagram

Description automatically generated

**Encoder?**

Each encoder consists of two sub-units, multi-head self-attention and feed forward neural network. After every subunit there is residual connection followed with a layer normalization shown in the left half of Fig 1.

**Decoder?**

The decoder also consists of the same sub-units and one more sub-unit that performs a masked multi-head attention to not allow it from attending to the words/data-points later in the series and prevent the information leak. The multi-head self-attention mechanism is also modified because it performs operation over the output from encoder as shown in right half of Fig 1.

**Scaled dot product Attention?**

Attention is a mechanism which try to depict how the human brain focuses on important thing. To understand this one standard example is the following sentence: **“The animal didn’t cross the street because it was too tired”**. For humans it is easy to understand the word “it” refers to the animal but how do you make the machines understand this? This is where the attention plays part.

To understand how the calculation in self-attention works let’s take an example of shorter sentence: “Come here please”. When processing the first word the steps are as follows:

1. First step is to calculate the Query, Key, and Value vector for each of these words embedding.
2. Second step is to calculate the dot product of query vector and the key vector
3. Third step is to divide the score by the square root of the dimension of the key (dk) which in the paper was 64, hence 8 here.
4. Calculate the SoftMax of the result from previous steps
5. Multiply the SoftMax score with the value vector
6. Point wise summation of value vector.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Word** | **Query** | **Key** | **Value** | **Score** | **Score / 8** | **Softmax** | **Softmax\*V** | **Sum** |
| Come | Q1 | K1 | V1 | Q1.K1 | Q1.K1/8 | S1 | S1\*V1 | Z1 |
| Here |  | K2 | V2 | Q1.K2 | Q1.K2/8 | S2 | S2\*V2 |  |
| Please |  | K3 | V3 | Q1.K3 | Q1.K3/8 | S3 | S3\*V3 |  |

Similar steps are done for the next two words and the attention vector is calculated and passed to the feed forward neural network. In the actual implementation this is done in matrix for better performance. Every word embedding is stacked to form a matrix and passed through and multiplying it with the weight matrix WQ, WK, WV to give us Q, K, V matrices. The attention is then calculated with the formula.

![Text

Description automatically generated]()

(Write about multi head attention)

References:

https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlp-state-of-the-art-models/

https://jalammar.github.io/illustrated-transformer/