ATTENTION IS ALL YOU NEED

- Archit Uniyal

MOTIVATION

To improve the results of sequence models, which are mainly constructed using RNNs, LSTMs or GRUs so that more accurate sequences can be obtained with less computational resources as well more sequence computations can be parallely performed.

IDEA

It is already known that RNNs and LSTMs using previous state h_{t-1} deduce the current hidden state h_t but RNNs and LSTMs fail in the case of longer sequence lengths because while training, sequential models precludes parallelisation i.e only the previous hidden state h_{t-1} is considered while training to deduce the next hidden state h_t. In order to resolve this problem transformer model is used which uses attention mechanism.

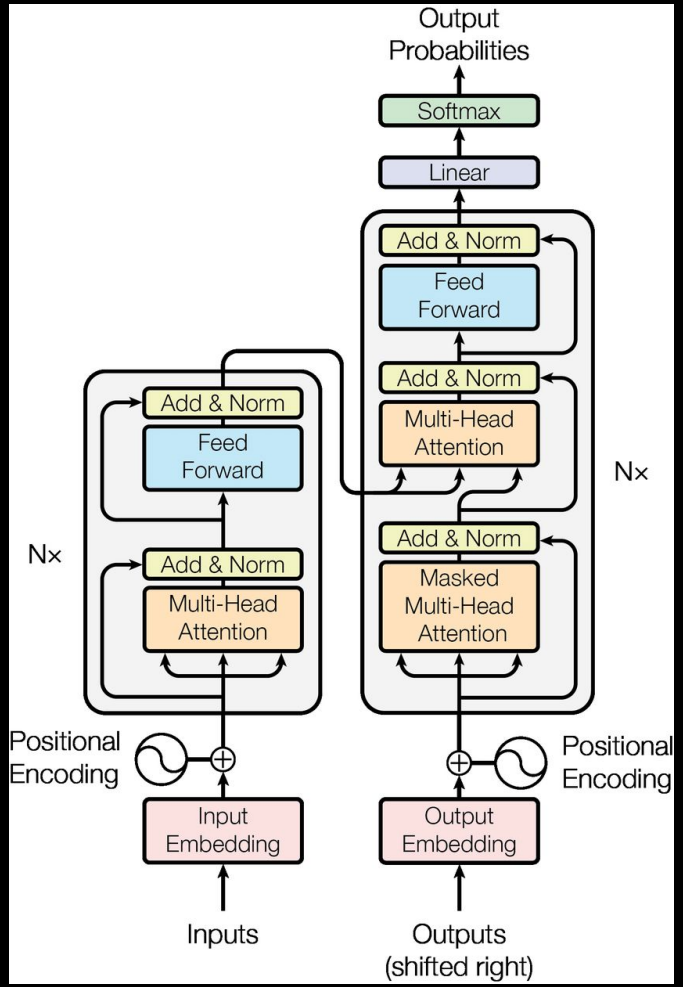
METHODOLOGY

N = 6

- Encoder and Decoder Stacks

Encoder : Each layer has 2 sub-layers, multi-head attention mechanism and a position wise FC feed-forward network. Residual connections are built between each sub-layer followed by normalisation.

Decoder : In addition to the two encoder sub-layers, the decoder consists of an additional layer, which performs multi-head attention over the encoder stack. The output embeddings are offset by one position to ensure that the predictions for position *i* are dependent only on the outputs of positions less than *i*.



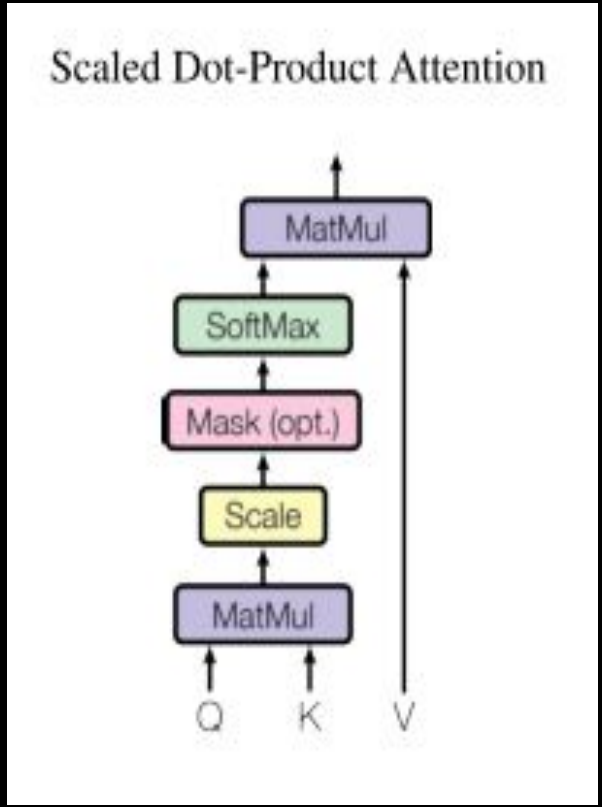
Cont...

- Scaled Dot-Product Attention

Inputs : Queries, keys of dimensions d_k and values of dimensions d_v.

$$\text{Attention}(Q, K, V) = \text{softmax}(Q.K^T / \sqrt{d_k}) V$$

The scaling factor $1/ \sqrt{d_k}$ is used to scale the dot-product attention since for bigger values of d_k the softmax will result in very small values of gradients.



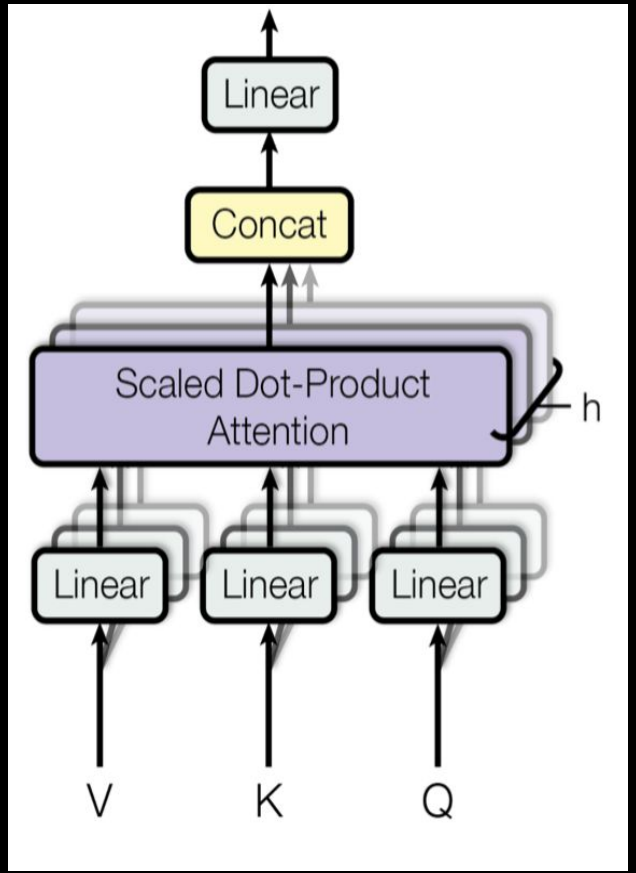
- Multi-Head Attention

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^o$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

Where the projections are parameter matrices $W_i^Q \in R^{d_{model} \times d_k}$,

$W_i^K \in R^{d_{model} \times d_k}$, $W_i^V \in R^{d_{model} \times d_v}$ and $W^O \in R^{hd_v \times d_{model}}$.



Positional Encoding

In order for the model to make use of the order of the sequence, information about the relative or absolute position of the tokens is injected into the sequence. It is done by adding positional encodings to the input embeddings at the bottom of the encoder and decoder stacks. The dimensions of positional embeddings is the same d_{model} as the embeddings so that they can be summed.

$$PE_{(pos,\ 2i)} = \sin(pos/1000^{2i\ /\ dmodel})$$

$$PE_{(pos,\ 2i+1)} = \cos(pos/1000^{2i\ /\ dmodel})$$

DISCUSSION POINTS

- Additive attention V/S Dot-product attention
- It is observed that in this paper RNNs and LSTMs aren’t used to generate sequences, instead an encoder-decoder architecture is used. How can we generate the same efficiency using RNNs or LSTMs?
- Transformers are considered better than recurrent or convolutional models.

THANK YOU