Attention becomes transformer

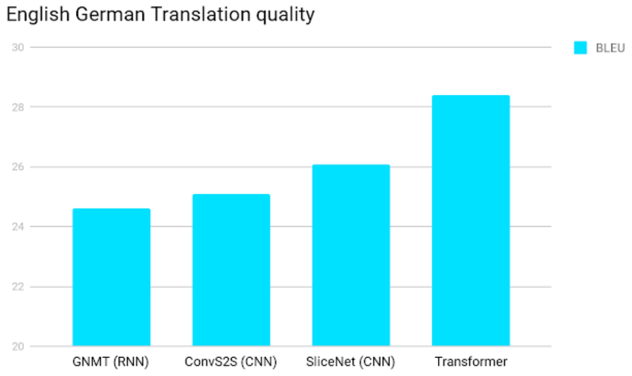
8 May, 2020

Alexey Zaytsev, head of Laboratory LARSS, PhD

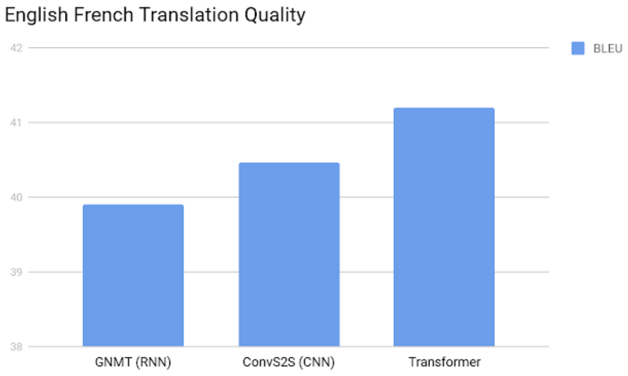
Foundations of Data Science

Skoltech
Skolkovo Institute of Science and Technology

New state of the art: **attention** is all we need



BLEU scores (higher is better) of single models on the standard WMT newstest2014 English to German translation benchmark.



BLEU scores (higher is better) of single models on the standard WMT newstest2014 English to French translation benchmark.

https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html

Skoltech
Skolkovo Institute of Science and Technology

Machine translation

Translate a sentence from one language to another

source language

target language

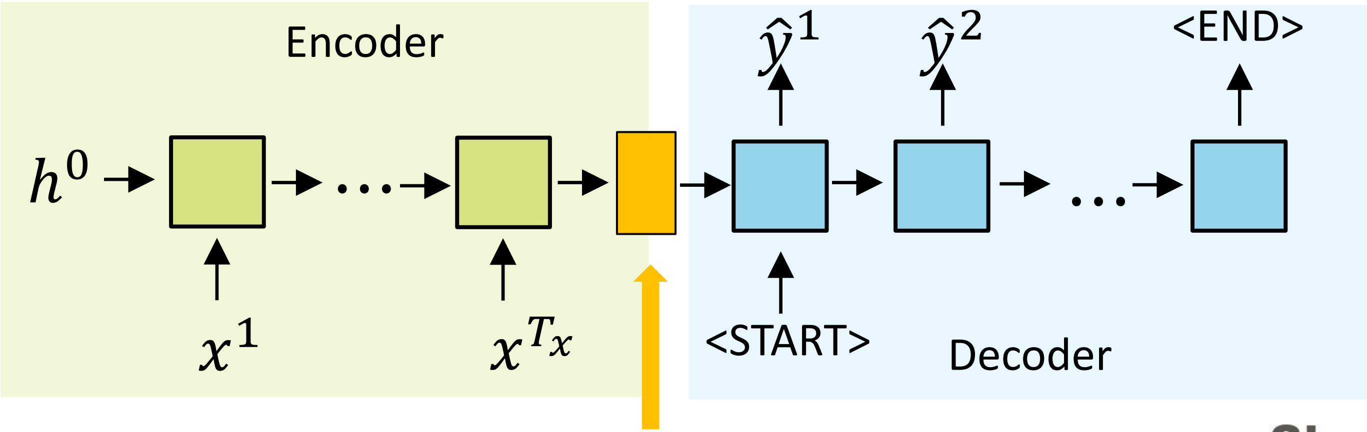
$\boldsymbol{x}$ → $\boldsymbol{y}$

A la guerre comme a la guerre

На войне как на войне

New world of seq2seq models

Summarization: long text – text summary
Dialogue: one phrase – another phrase
Parsing: input – output parse as a sequence
Code generation: task description – python code



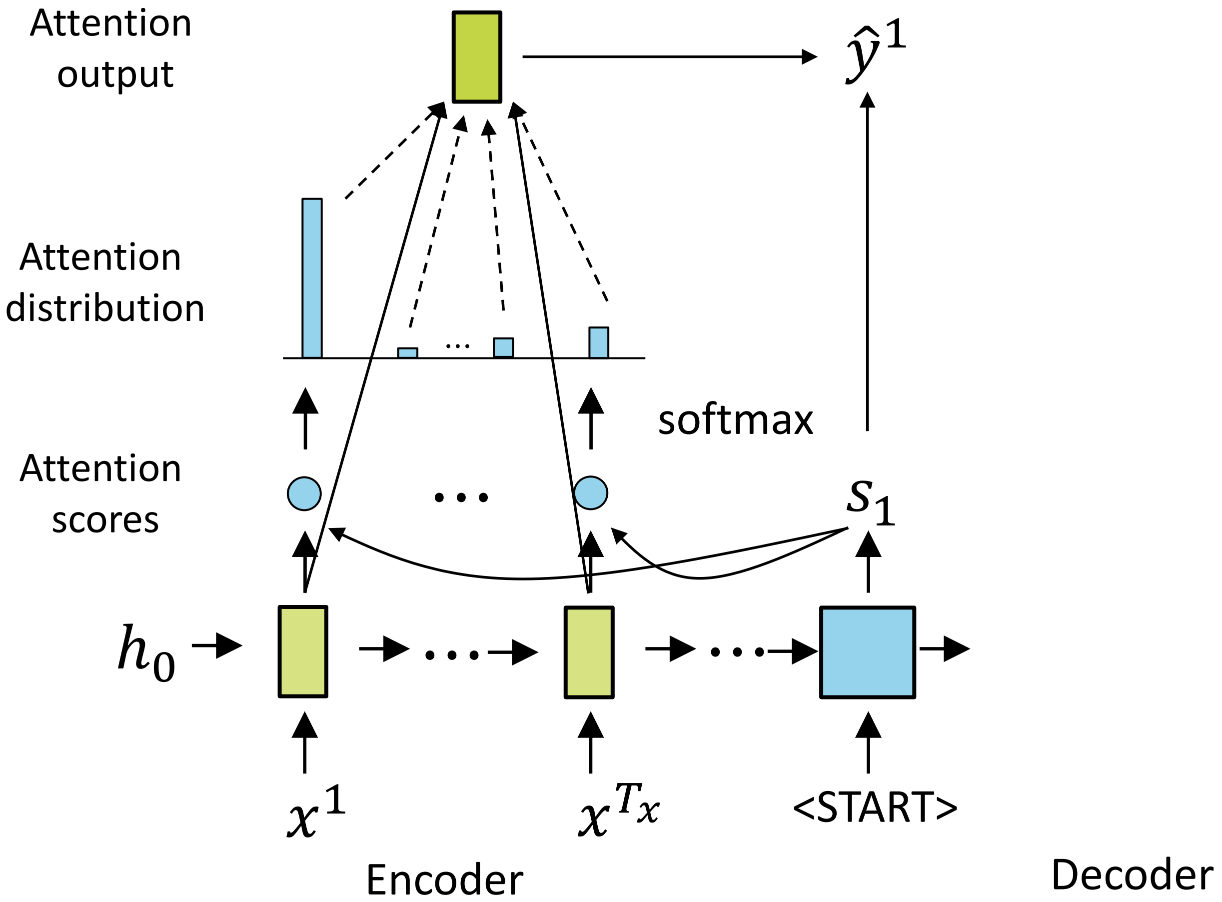
All information about the sequence is in this vector

Skoltech
Skolkovo Institute of Science and Technology

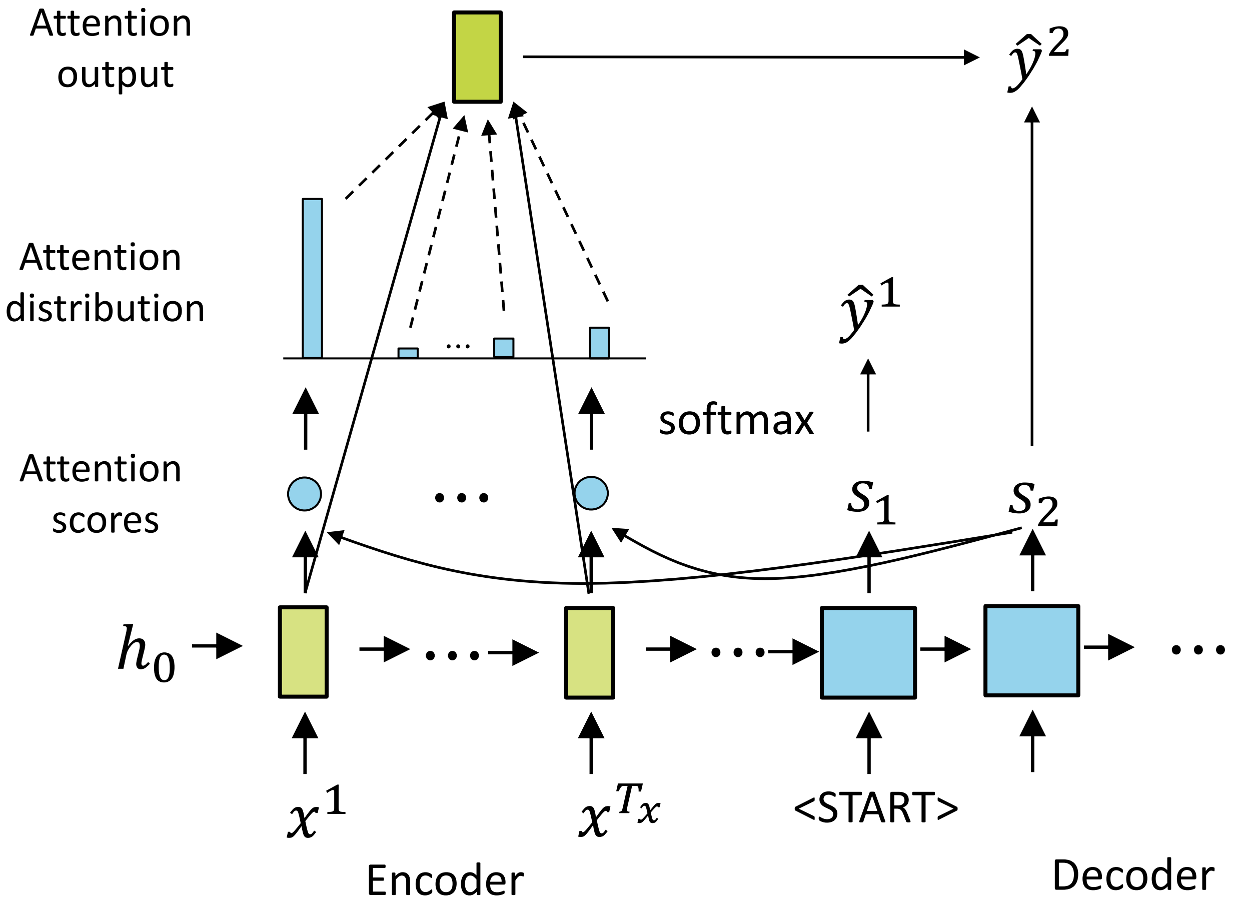
Attention

- Solution to the bottleneck problem
- Direction connection between parts of input and output sequence

Sequence 2 sequence with attention



Sequence 2 sequence with attention



Attention: formulas

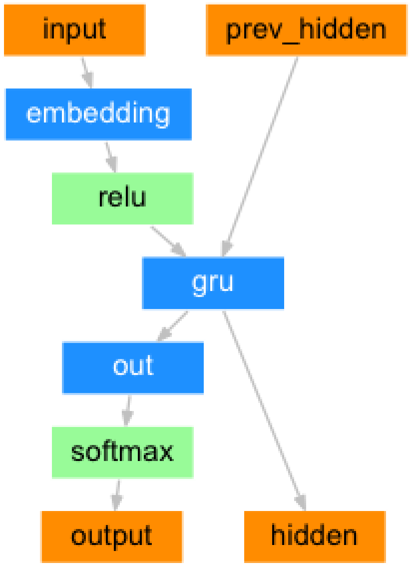
- First RNN produces encoder hidden states $\boldsymbol{h}_1, \dots, \boldsymbol{h}_{T_x} \in \mathbb{R}^h$
- Decoder hidden state $\boldsymbol{s}_t \in \mathbb{R}^h$ at time step t
- Attention scores for step t:
  $\boldsymbol{e^t} = [\boldsymbol{s}_t^{\boldsymbol{T}}\boldsymbol{h}_1, \dots, \boldsymbol{s}_t^{\boldsymbol{T}}\boldsymbol{h}_{T_x}] \in \mathbb{R}^{T_x}$
- Softmax to get attention distribution: all values are positive, sum of all values is 1:
  $\boldsymbol{\alpha^t} = \mathrm{softmax}(\boldsymbol{e^t}) \in \mathbb{R}^{T_x}$
- Attention output $\boldsymbol{a}_t$ is the weighted sum of hidden states:

$$\boldsymbol{a}_t = \sum_{i=1}^{T_x} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$

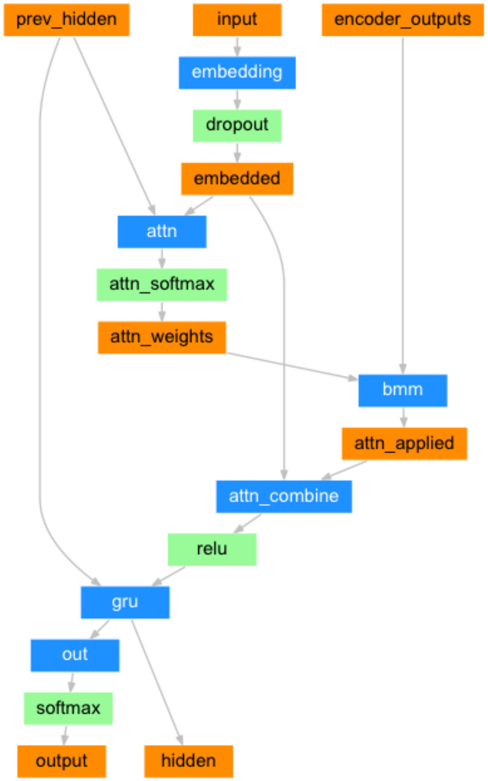
- We concatenate the attention output $\boldsymbol{a}_t$ with the decoder hidden state $\boldsymbol{s}_t$ and proceed to the non-attention part of our seq2seq model

$$[\boldsymbol{a}_t, \boldsymbol{s}_t] \in \mathbb{R}^{2h}$$

Attention: blocks



Simple decoder



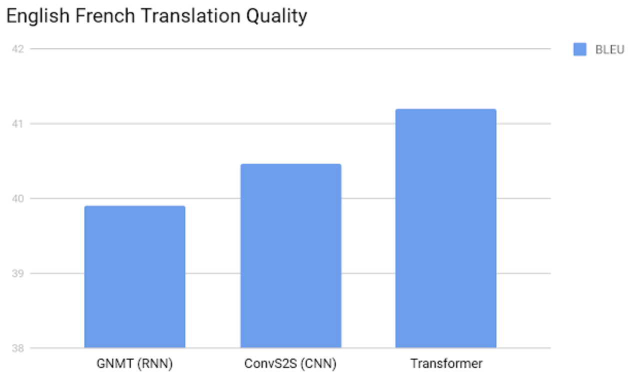
Decoder with attention

https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html

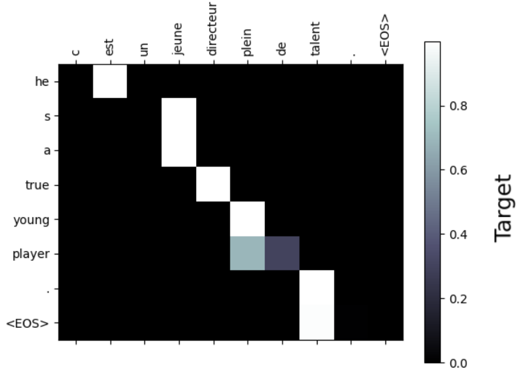
Attention is just great

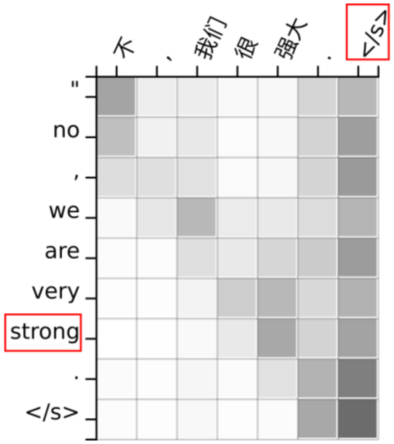
Similar to RNN seq2seq, but greater!

- Significantly improves performance of NMT
- Solves the bottleneck problem
  - All encoder tokens are connected to all decoder tokens
- No more vanishing gradients
  - All to All connection
- Provides some interpretability
  - see alignment figure



BLEU scores (higher is better) of single models on the standard WMT newstest2014 English to French translation benchmark.





https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html

Skoltech
Skolkovo Institute of Science and Technology

Attention is a general deep learning idea

We can use attention in many architectures and many tasks

- Other NLP problems
- Graph Neural networks

Key value interpretation:

s_i - query to a database | Hidden state of the decoder

k_j - keys in the database | Hidden state of the encoder

h_j - values in the database | Hidden state of the encoder

We calculate correspondences $e(s_i, k_i)$

Then we extract the attention as weighted sum of values $\mathbf{a}_{\mathrm{i}} = \sum_{j=1}^{T_x} \alpha_j \boldsymbol{h}_j$

Skoltech
Skolkovo Institute of Science and Technology

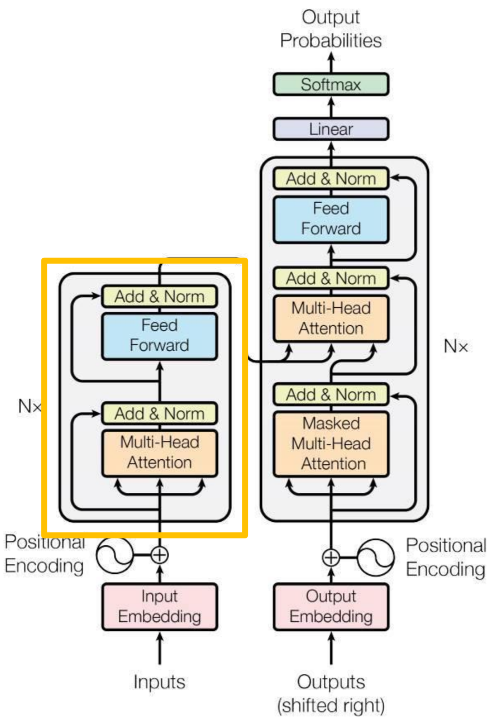
Transformer is based on the same idea

Now we completely drop RNN part

Also we repeat self-attention many times

Further we consider a separate block:

- Multihead attention
- Feed Forward



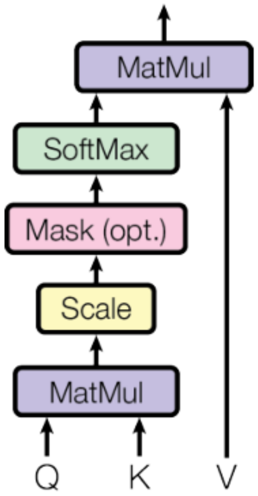
Self-attention block

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

d_k is the dimension of query and key,
we scale to take control of large values of dot-product in high dimensions

A possible option is to replace dot-product used here with a single-hidden layer neural network.

Scaled Dot-Product Attention

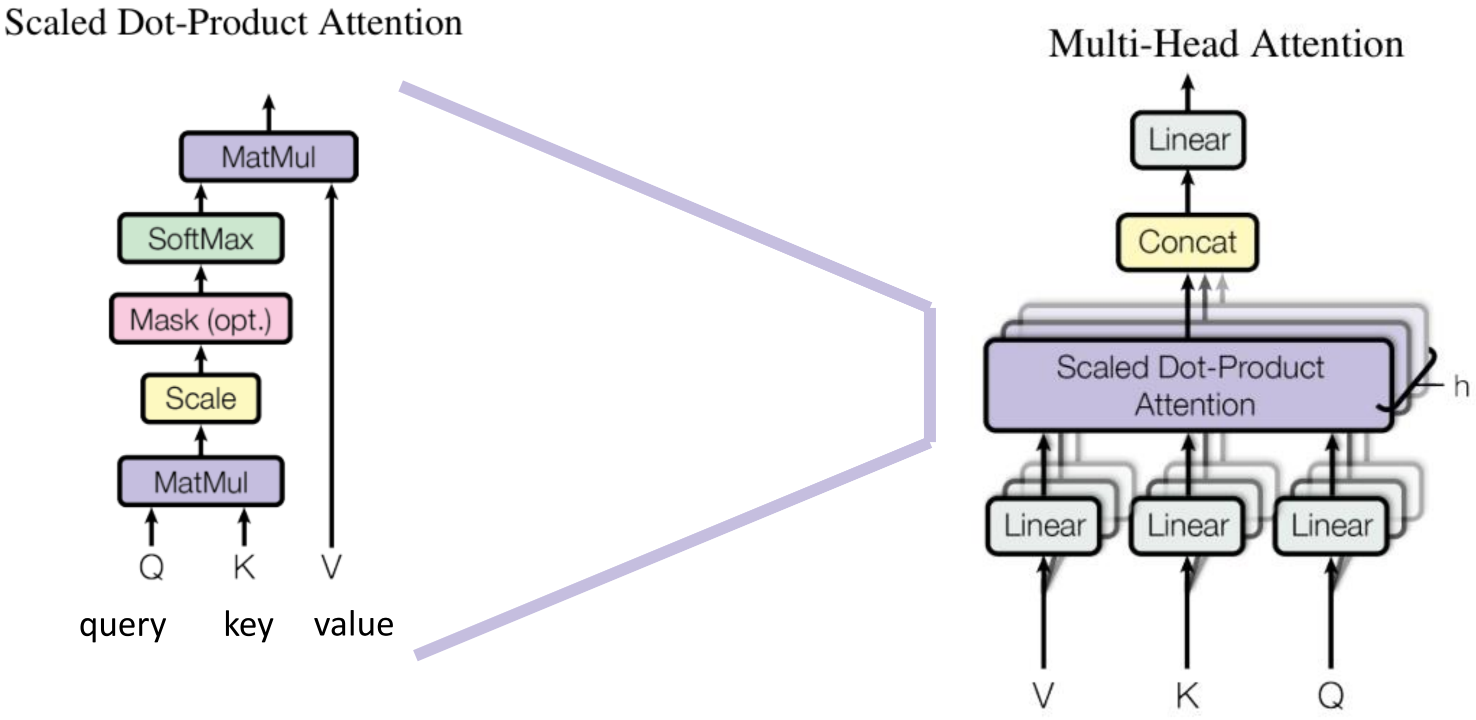


query key value

“Attention is all you need” paper

Multi-Head attention

$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head_1}, \ldots, \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



"Attention is all you need" paper

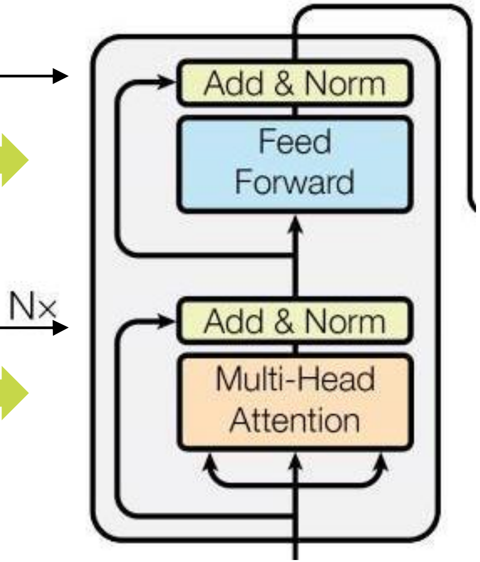
Full block

$\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$

Two linear transformation with ReLU activation in between

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



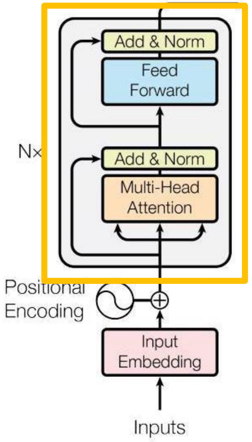
“Attention is all you need” paper

Position encoding

In addition to usual embeddings of inputs we use position encoding to capture position

They are not one-hot vectors, as we want to handle various-length sequences

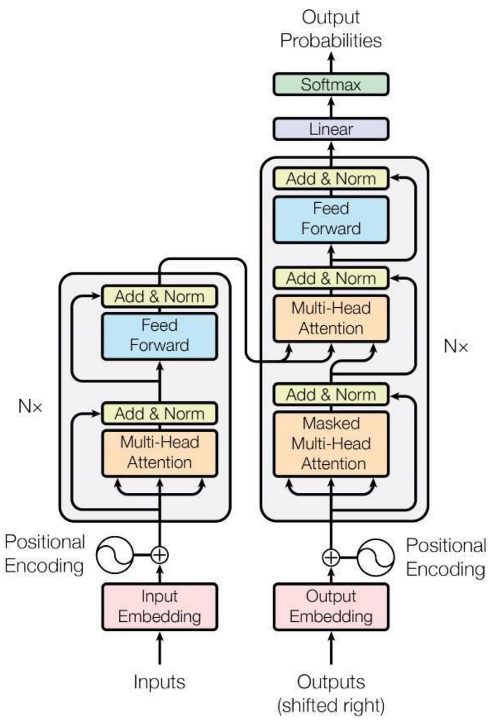
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



Decoder is similar

It has additional sublayer to take into account attentions from encoder

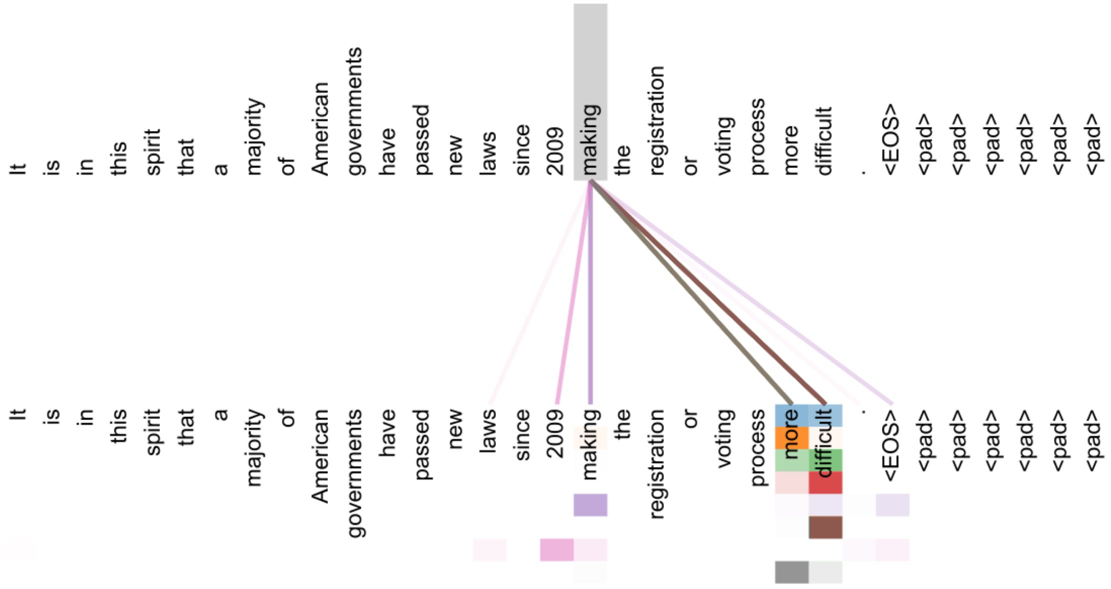
We generate one token and proceed to the next token generation



“Attention is all you need” paper

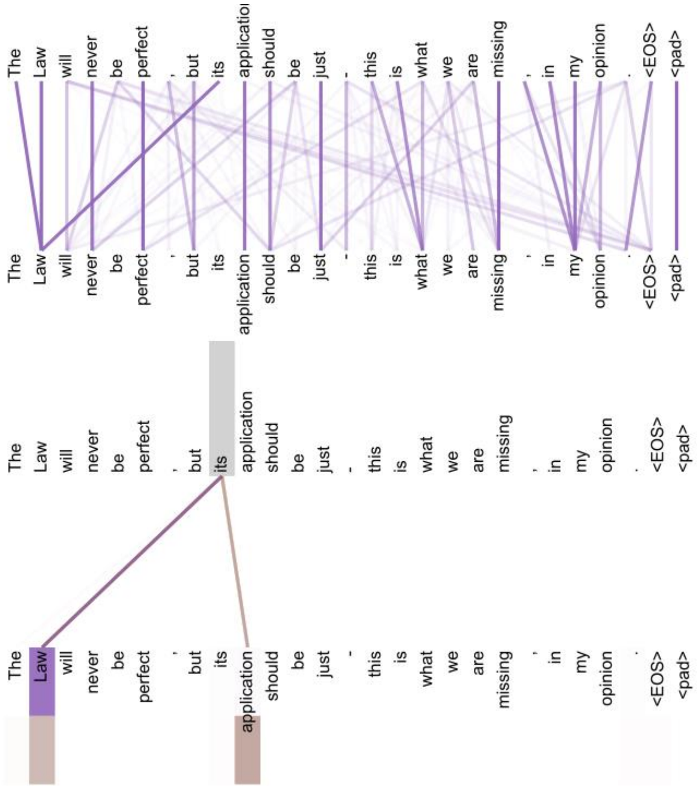
Skoltech
Skolkovo Institute of Science and Technology

Attention visualizations



“Attention is all you need” paper

Attention visualizations



“Attention is all you need” paper

Sources

“Attention is all you need” paper

Skoltech

Skolkovo Institute of Science and Technology