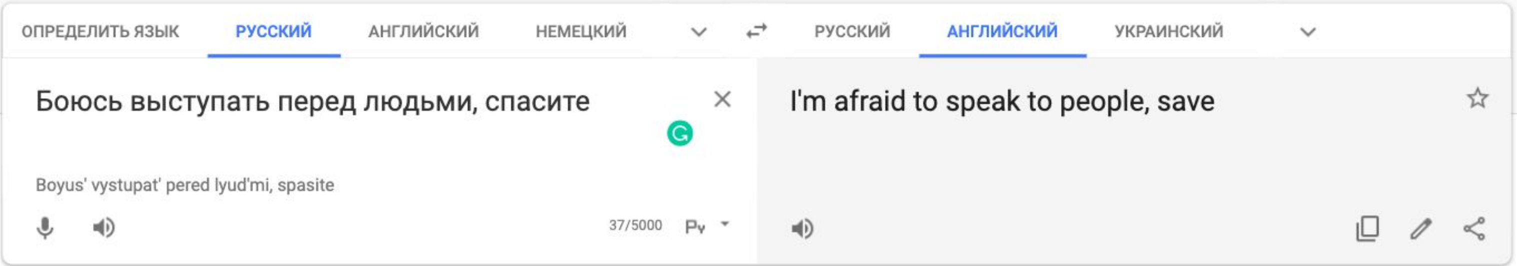
# Attention Is All You Need

Rak Arina, AMI171
HSE Research Seminar
2019

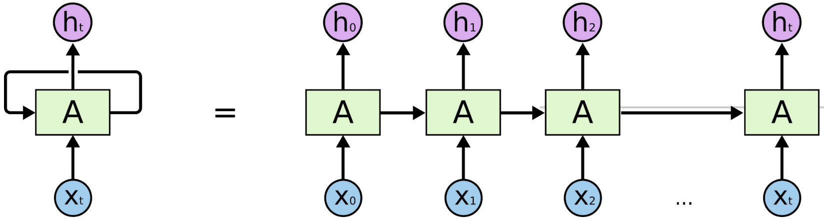
**Sequence transduction** — transformation of input sequences into output sequences:

- Machine translation
- Speech recognition
- Spelling correction
- Part of speech tagging



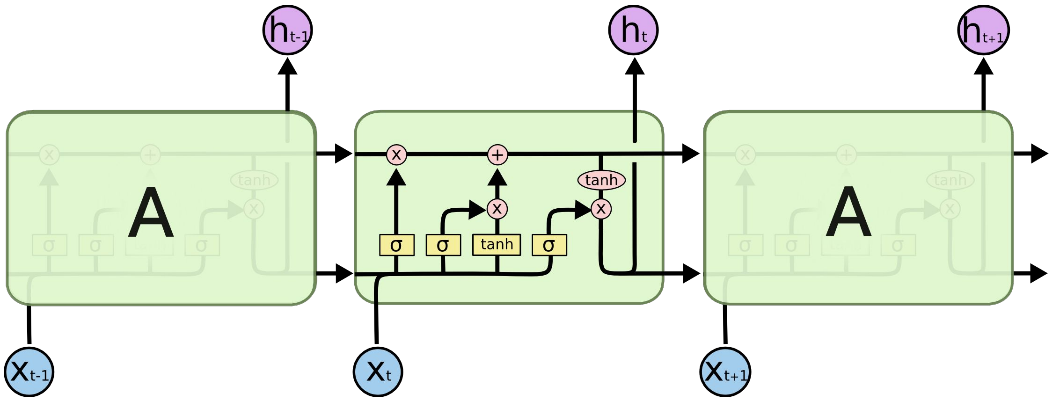
# Recap | RNNs

**Vanilla RNN**

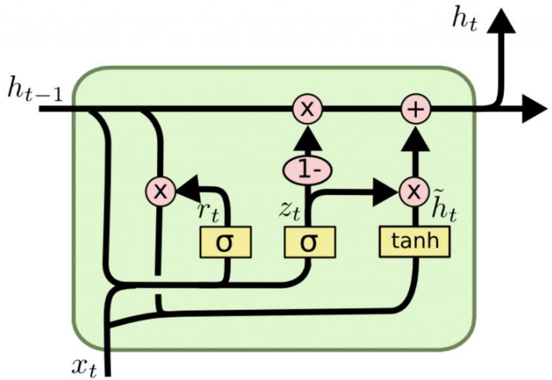



**Problems:**

- Learns slow
- Vanishing/exploding gradients
- Difficult to learn dependencies between distant positions

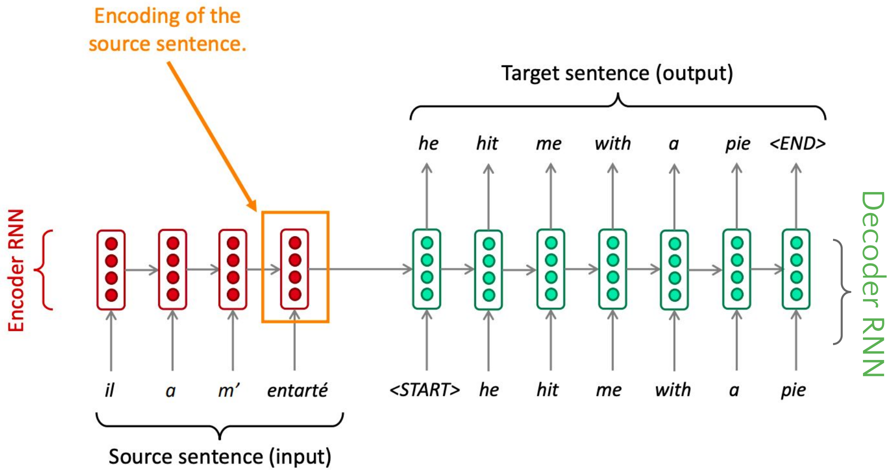



**LSTM**

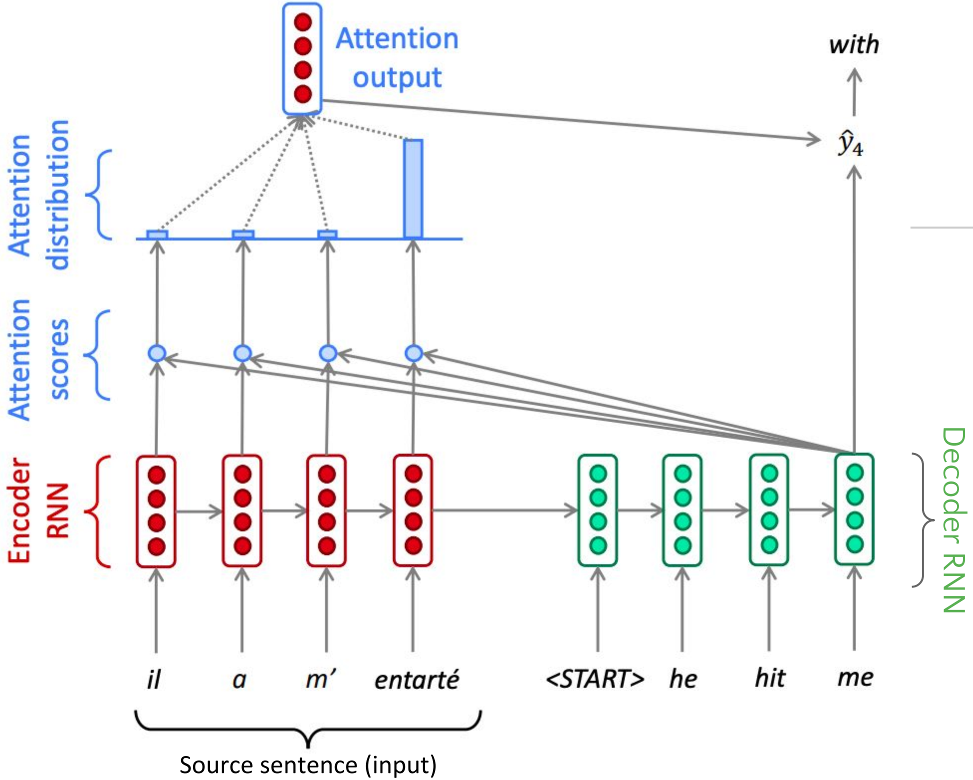



**GRU**

# Recap | Attention




**Classical seq2seq**




**Attention seq2seq**

**Attention:**

- Improves performance
- Helps with vanishing gradients
- Solves the bottleneck problem
- Helps with interpretability

**BUT:** Models get more and more complex and the computations still can not be done in parallel => **SLOW**

# Transformer

Was proposed in 2017 by Google

In WMT (MT conference + competition):

The summary report in 2016 contains the word ‘RNN’ 44 times

The summary report in 2018 contains the word ‘RNN’ 9 times and the word ‘Transformer’ 63 times

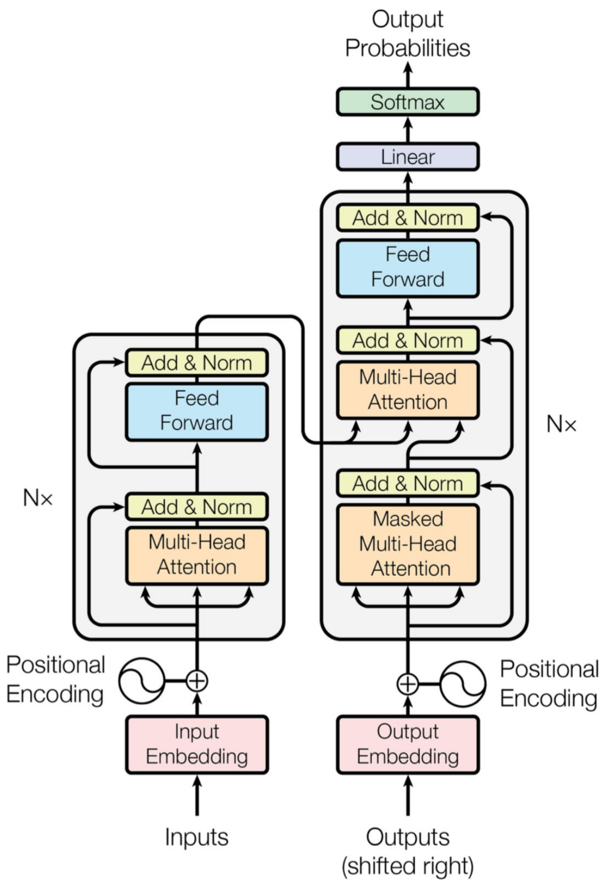
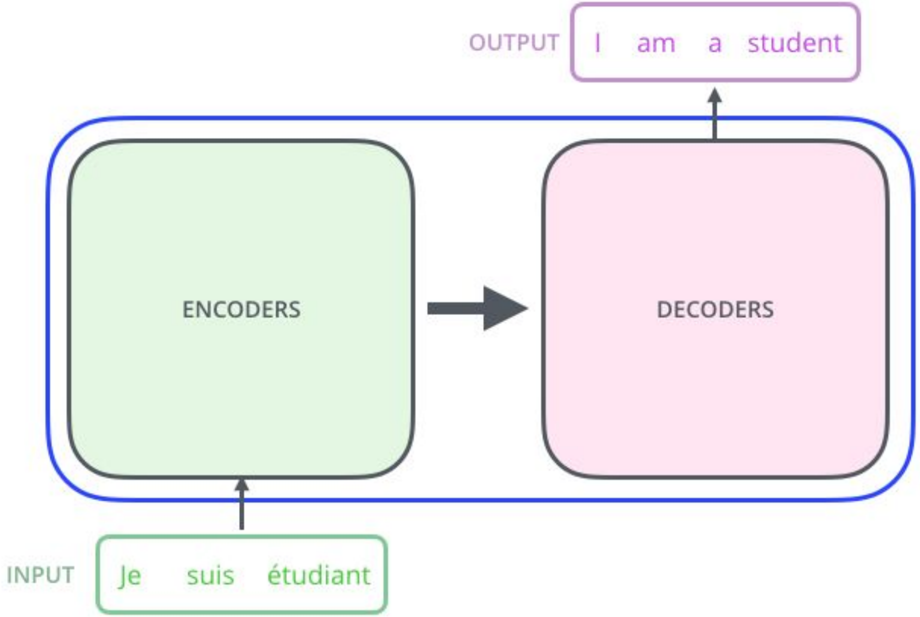



Figure 1: The Transformer - model architecture.

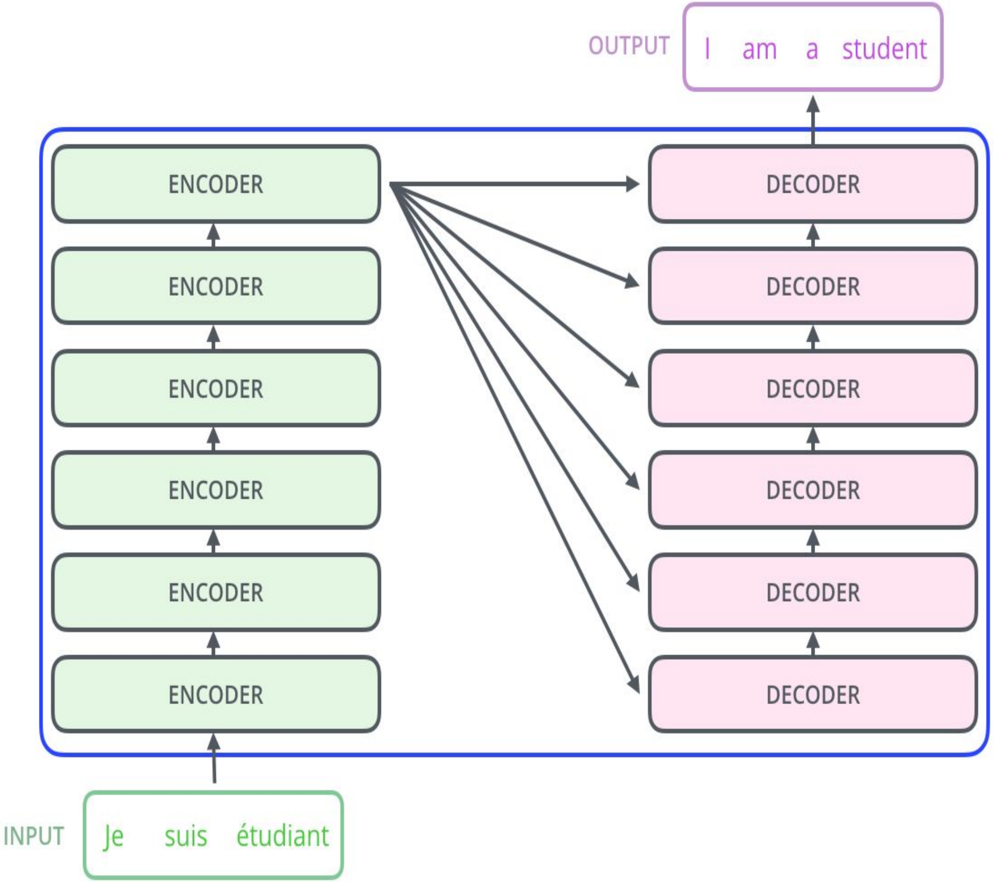
## Architecture

Encoder receives a list of fixed size of vectors each of the size of the embeddings dimensionality

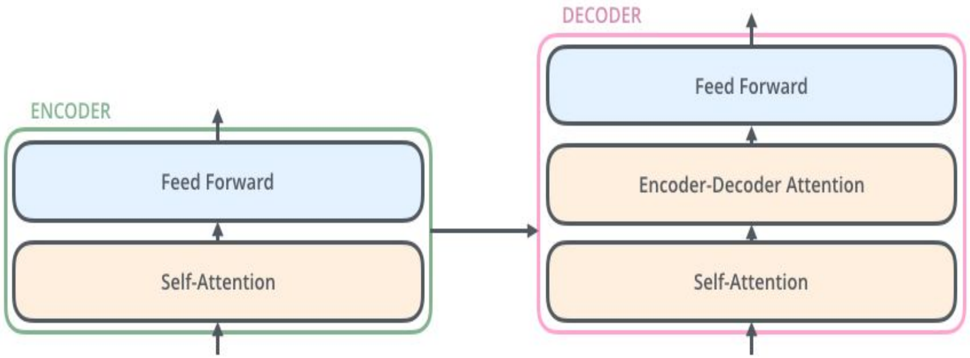


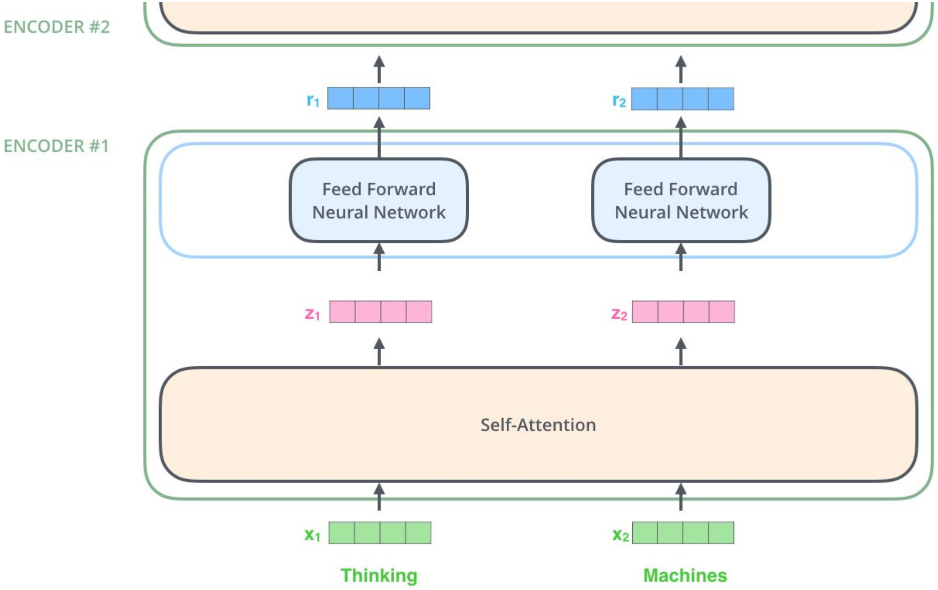
## Architecture

All Encoder and Decoder blocks have the same architecture, but they do not share weights.



Self-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence.



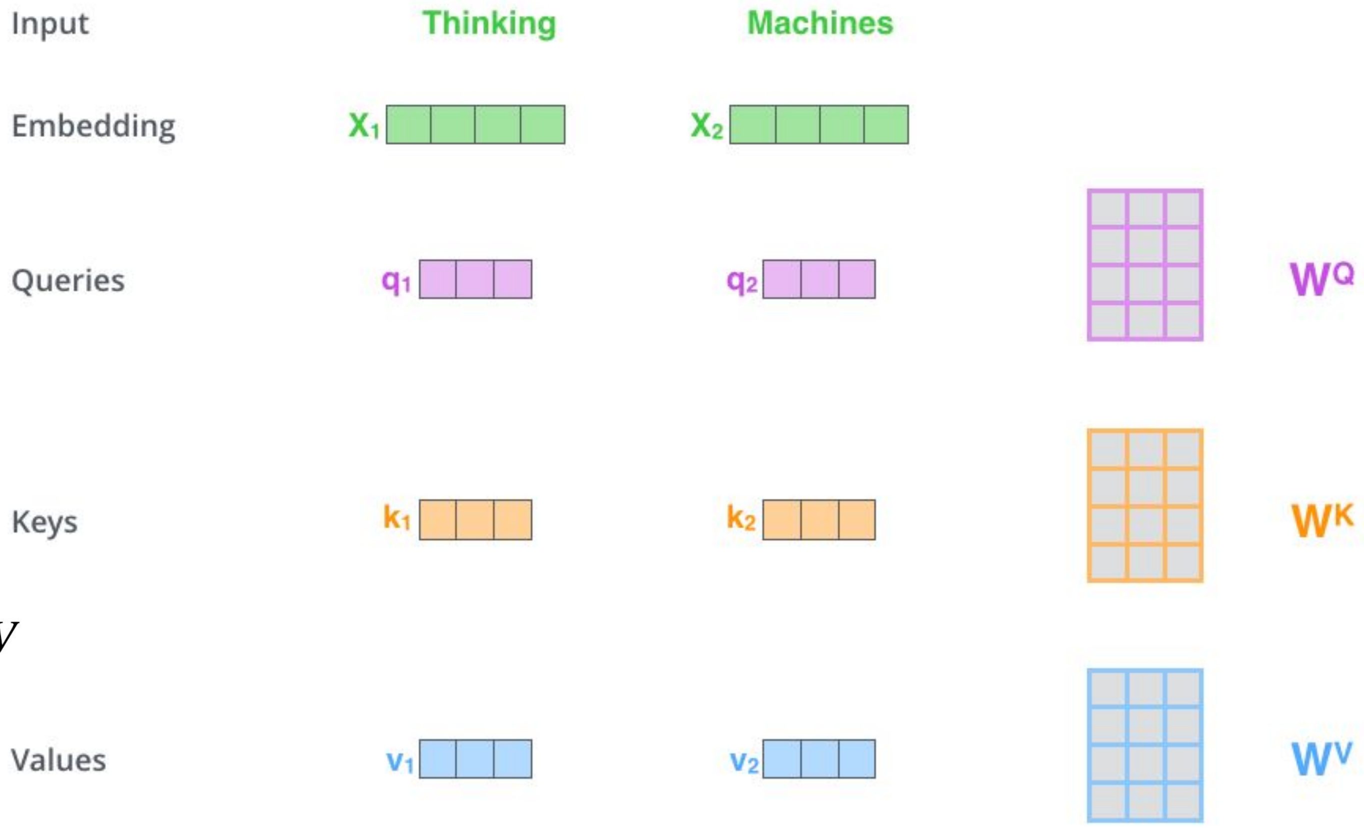




Self-attention in Encoder and Decoder are the same, except for the fact that Decoder can look only on the words previous to the current one

# Self-Attention

For each word x a query q, a key k and a value v is calculated by multiplying x by matrices

$W^Q$, $W^K$ and $W^V$ respectively



# Self-Attention

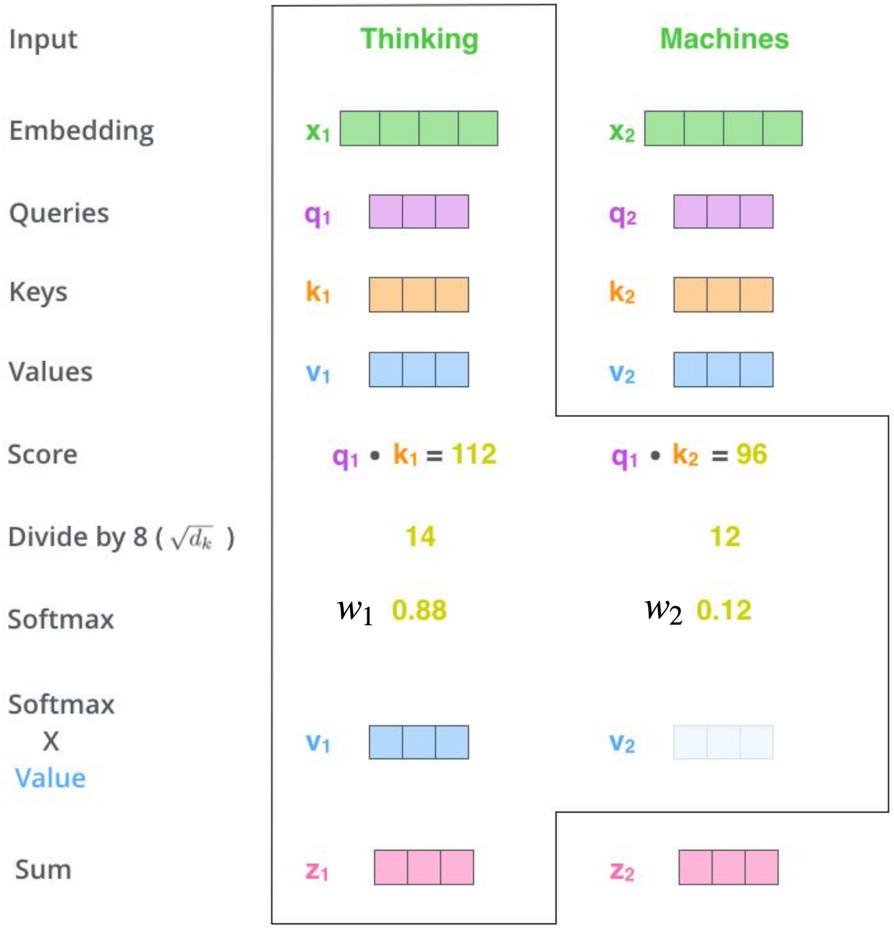
For every $x_i$

$< q_i, k_j >, \ j \in \{1, \ldots, input\_len\}$

is calculated, which is a score reflecting the weight of $v_j$ in the representation of $x_i$

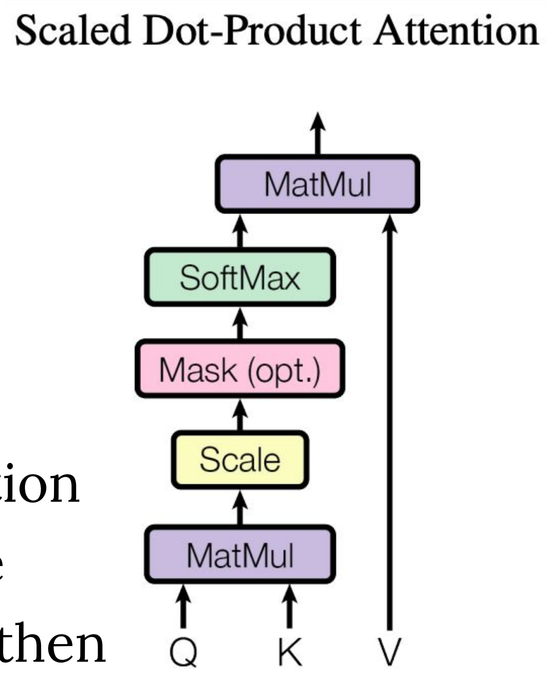
$$z_i = \sum_{j=1}^{input\_len} w_j v_j$$

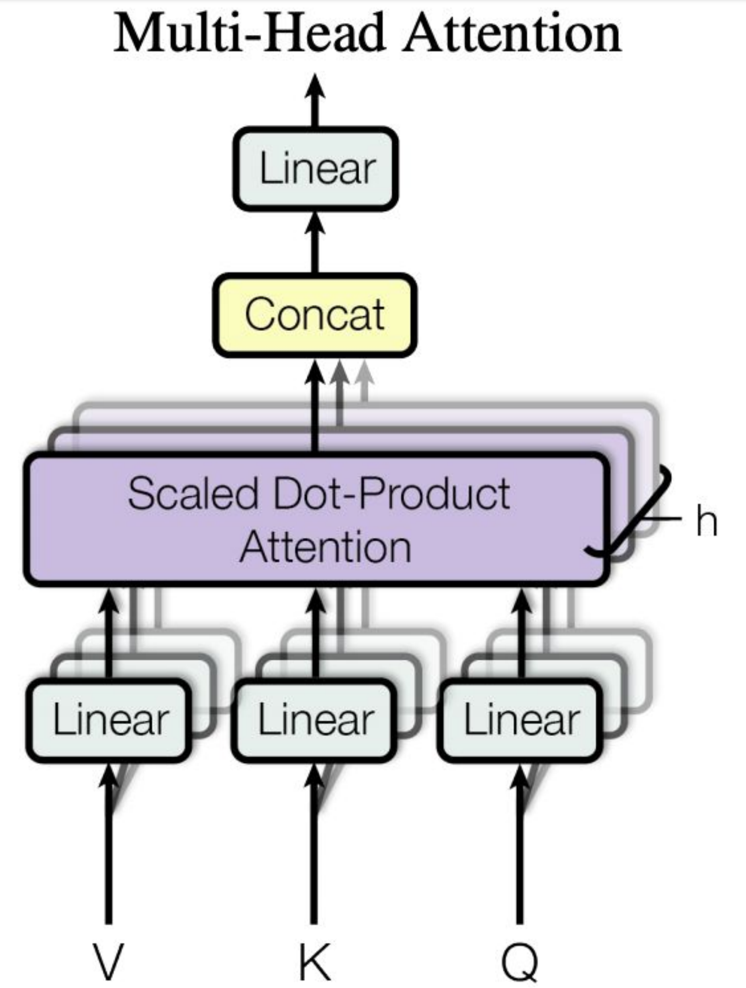
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



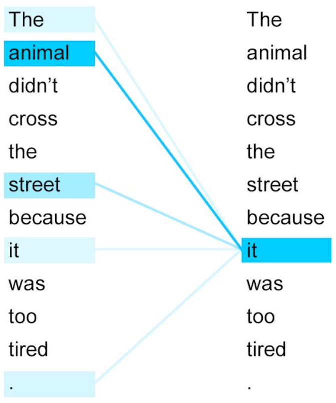
# Self-Attention

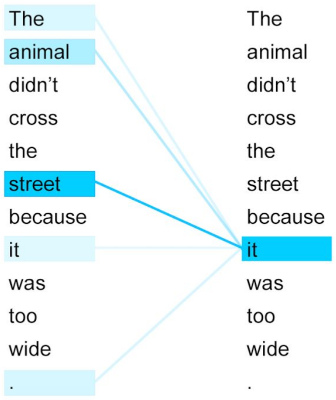
Multi-Head Attention just does the same thing h times and then concatenates the results and projects it back to the dimension of x with a linear layer

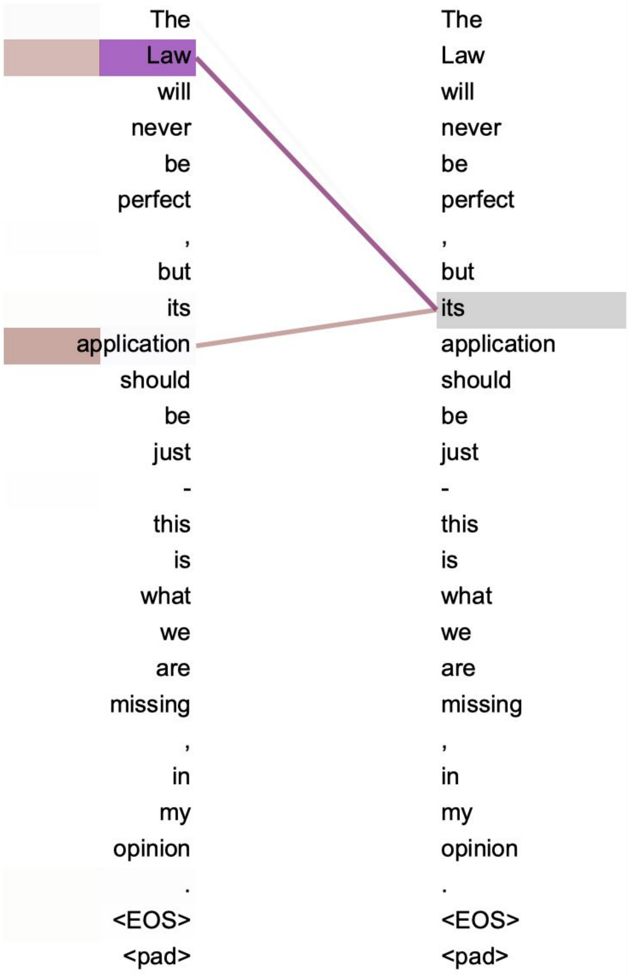


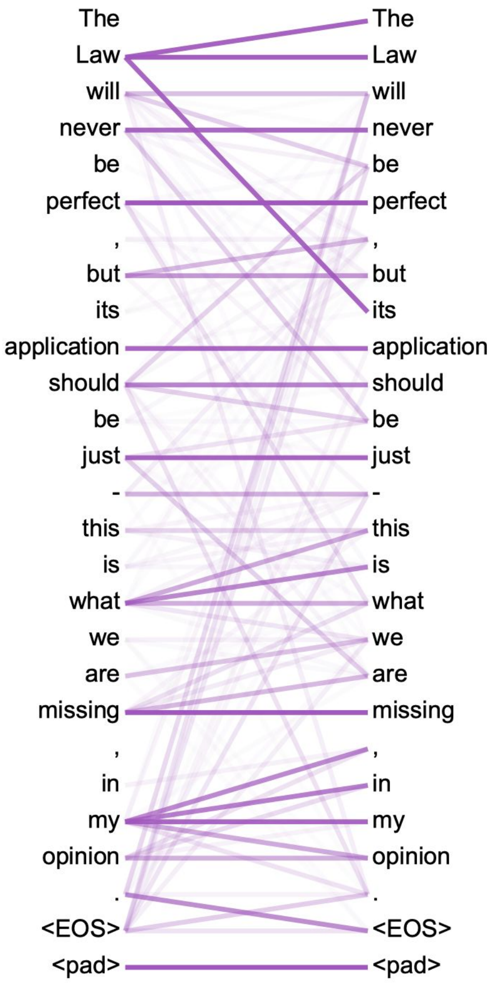



# Self-Attention examples



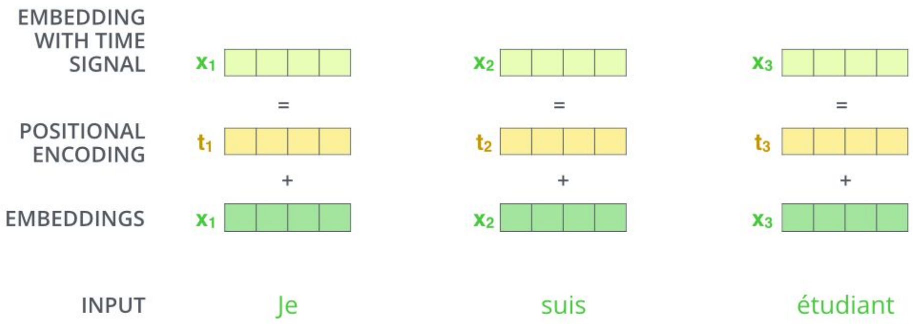



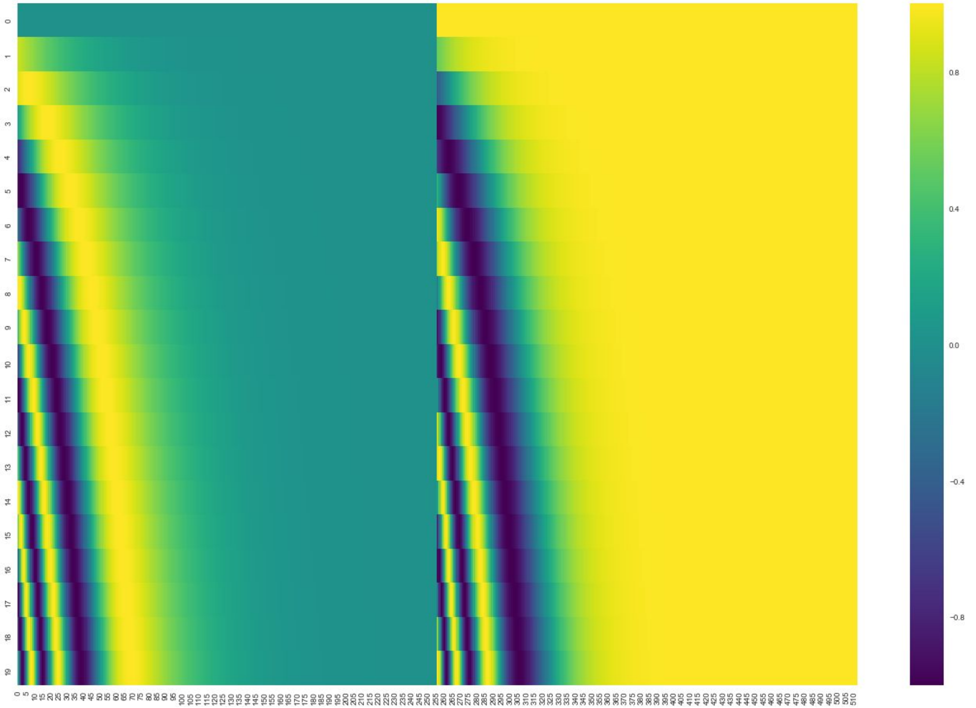



# Positional encoding

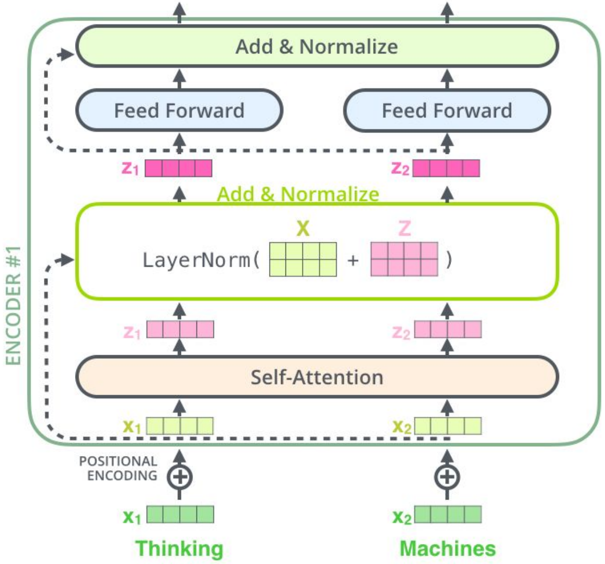
In order for the model to make use of the order of the sequence, some information about the relative or absolute position of the tokens of the sequence must be injected

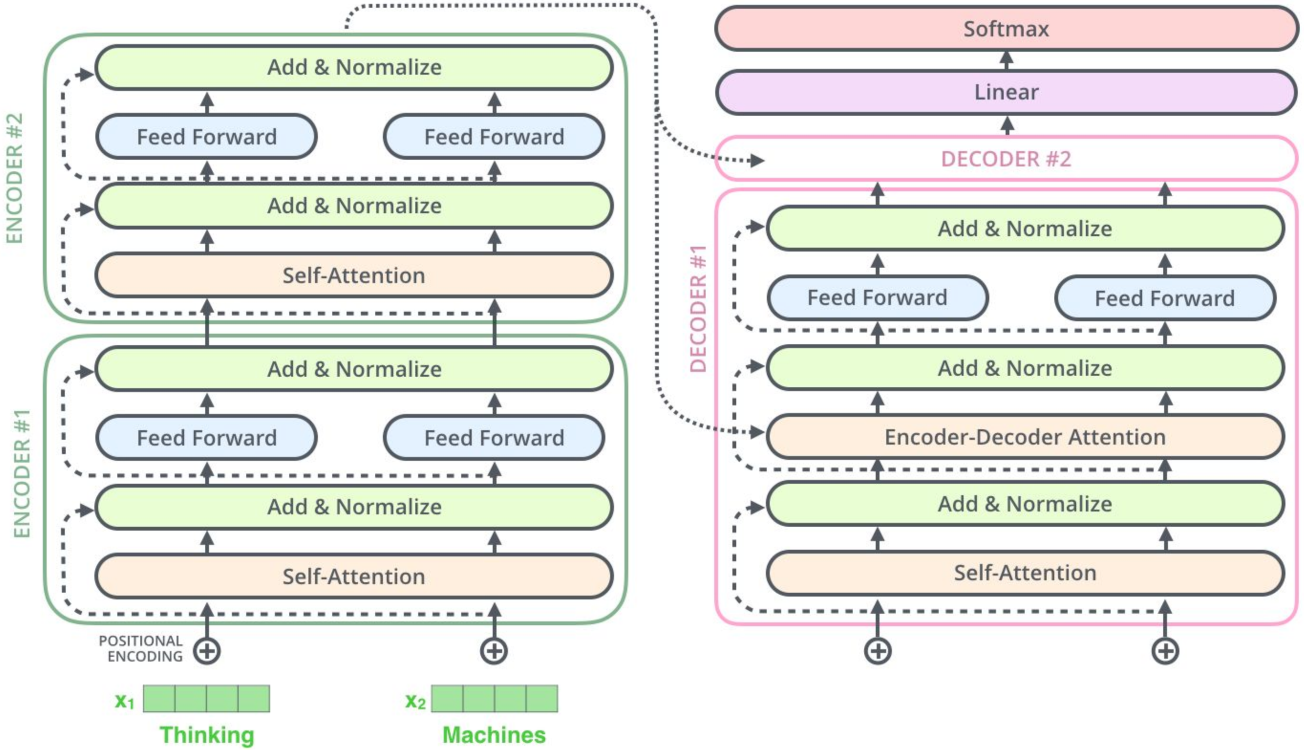
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



# Architecture



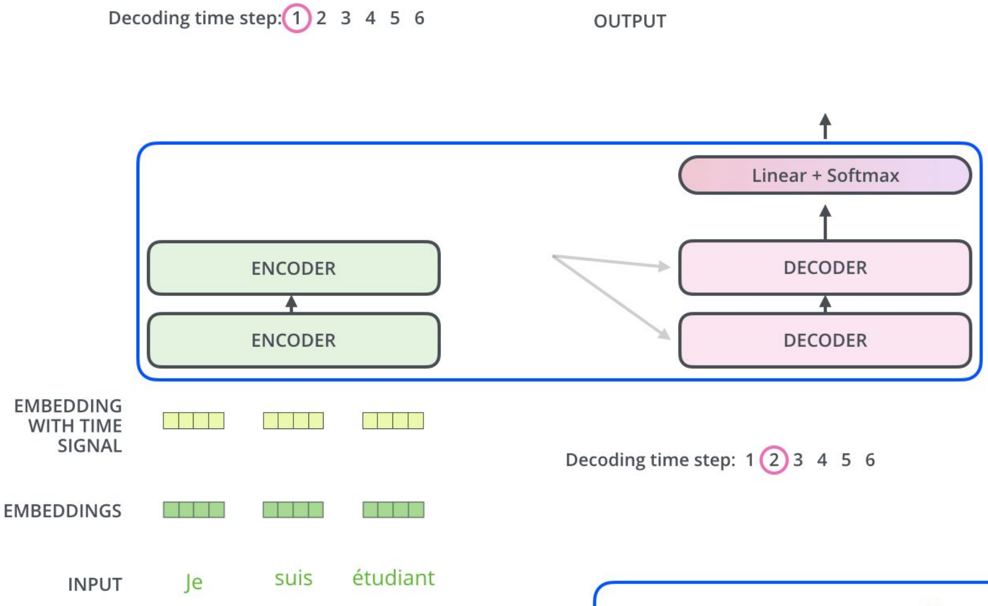




There are skip connections around each of the sublayers of the blocks and layer normalization after each sublayer.

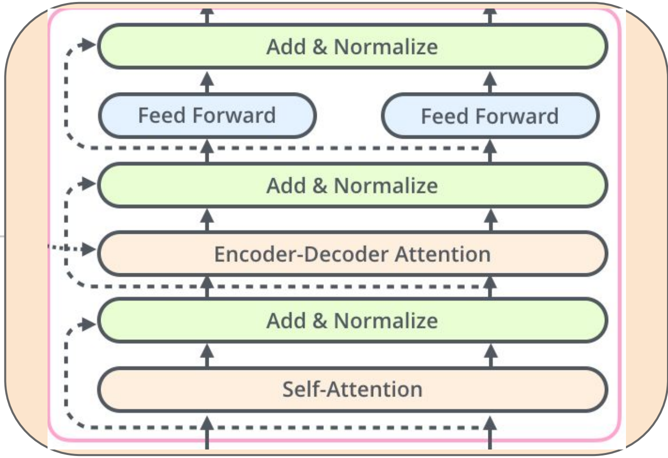
| Layer Type | Complexity per Layer | Sequential Operations | Maximum Path Length |
|---|---|---|---|
| Self-Attention | $O(n^2 \cdot d)$ | $O(1)$ | $O(1)$ |
| Recurrent | $O(n \cdot d^2)$ | $O(n)$ | $O(n)$ |
| Convolutional | $O(k \cdot n \cdot d^2)$ | $O(1)$ | $O(log_k(n))$ |
| Self-Attention (restricted) | $O(r \cdot n \cdot d)$ | $O(1)$ | $O(n/r)$ |

Motivating our use of self-attention we consider three desiderata.

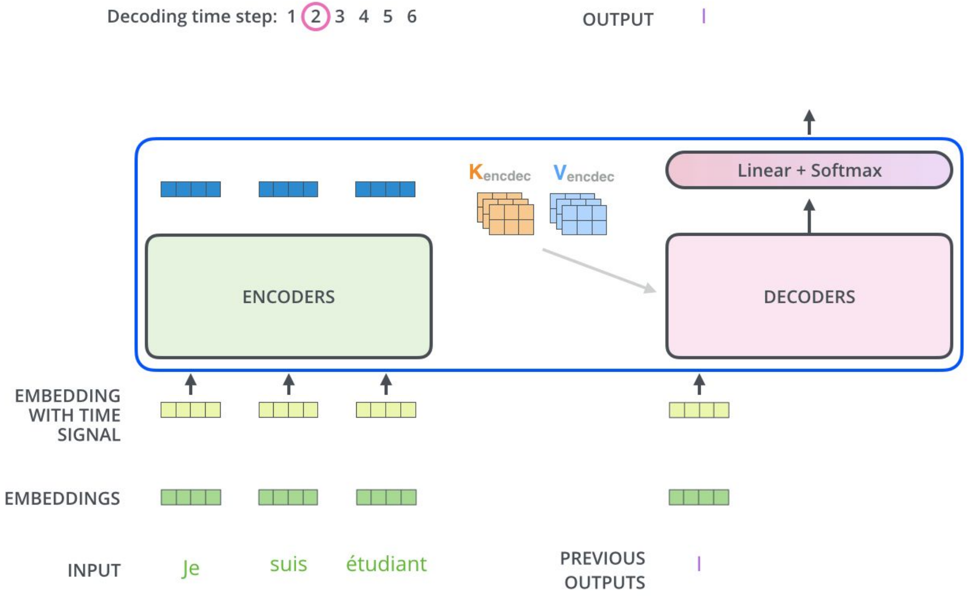
- One is the total computational complexity per layer.
- Another is the amount of computation that can be parallelized, as measured by the minimum number of sequential operations required.
- The third is the path length between long-range dependencies in the network. Learning long-range dependencies is a key challenge in many sequence transduction tasks.



Decoding time step: 1 2 3 4 5 6
OUTPUT
Linear + Softmax
ENCODER
ENCODER
DECODER
DECODER
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant

Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Encoder-Decoder Attention
Add & Normalize
Self-Attention

Decoding time step: 1 2 3 4 5 6
OUTPUT
I
Kencdec
Vencdec
Linear + Softmax
ENCODERS
DECODERS
EMBEDDING WITH TIME SIGNAL
EMBEDDINGS
INPUT
Je
suis
étudiant
PREVIOUS OUTPUTS
I

# Training

Untrained Model Output

| a | am | I | thanks | student | <eos> |
|---|---|---|---|---|---|
| 0.2 | 0.2 | 0.1 | 0.2 | 0.2 | 0.1 |

Correct and desired output

| a | am | I | thanks | student | <eos> |
|---|---|---|---|---|---|
| 0.0 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 |

**Target Model Outputs**

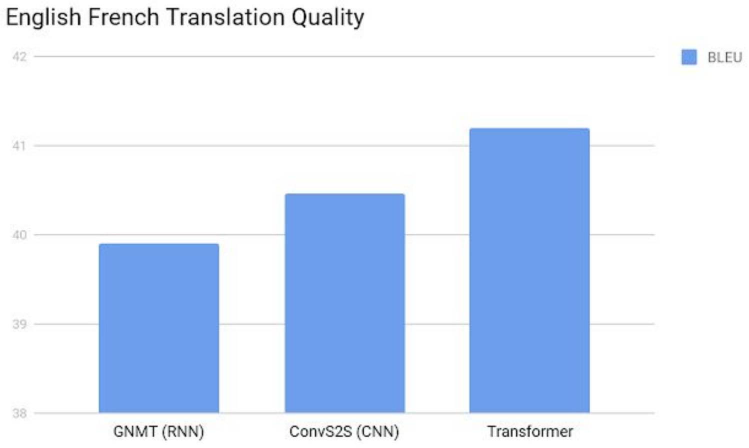
| Output Vocabulary: | a | am | I | thanks | student | <eos> |
|---|---|---|---|---|---|---|
| position #1 | 0.0 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 |
| position #2 | 0.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| position #3 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| position #4 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0.0 |
| position #5 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |

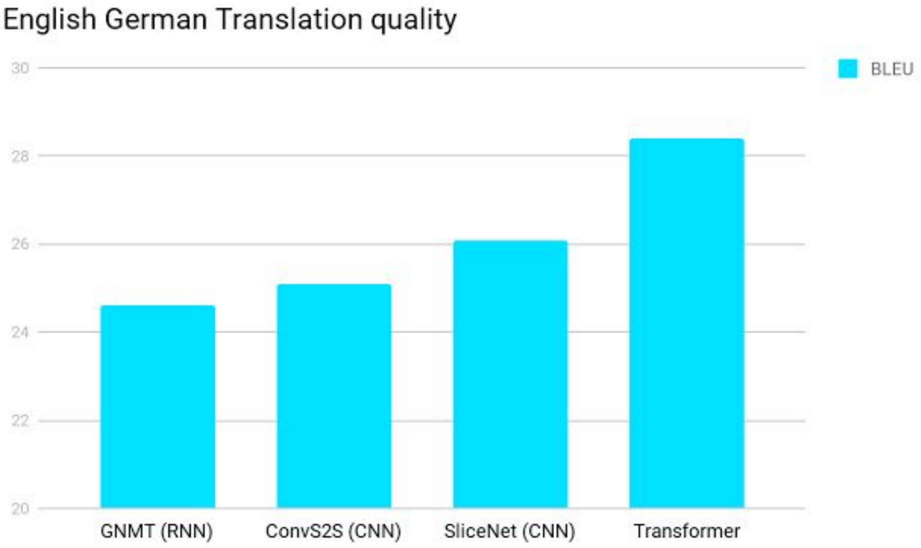
The output is a probability distribution.

Loss:

- Cross-entropy: $-\sum_i y\_true_i log\,(p\_pred_i)$
- KL - divergence: $-\sum_i p\_true_i log\left(\frac{p\_true_i}{p\_pred_i}\right)$

# Results in 2017




Transformers are used in both
Google and Yandex Translate

# Вопросы

- Что подается на вход Encoder-у трансформера?
- Что такое q, k, v в слое self-attention?
- Чем отличаются слои self-attention у Encoder-a и Decoder-a?

# Resources

- https://arxiv.org/pdf/1706.03762.pdf (original paper)
- https://habr.com/ru/post/341240/ (less papers but in Russian)
- http://jalammar.github.io/illustrated-transformer/ (best pics and explained quite nicely)
- https://www.youtube.com/watch?v=S0KakHcj_rs&t=1132s (video on the paper)
- https://www.youtube.com/watch?v=QEw0qEa0E50&feature=youtu.be (CS224n, Stanford's course on NLP)
- https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html