Attention Is All You Need

2017

Rak Arina
HSE Research Seminar
2022

Plan

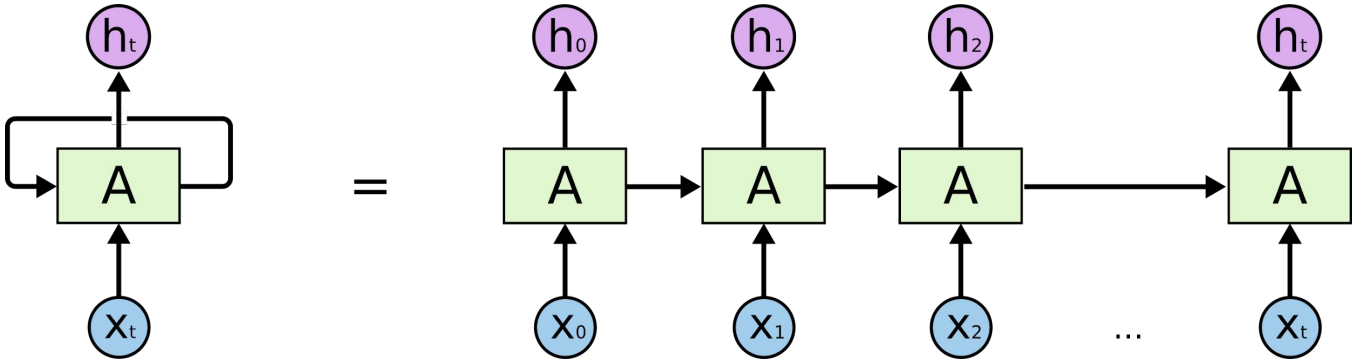
- Intro
- Recap: Autoregressive models and attention
- Architecture:
  - Encoder
  - Self-attention
  - Decoder
  - Positional embeddings
- Questions

Task

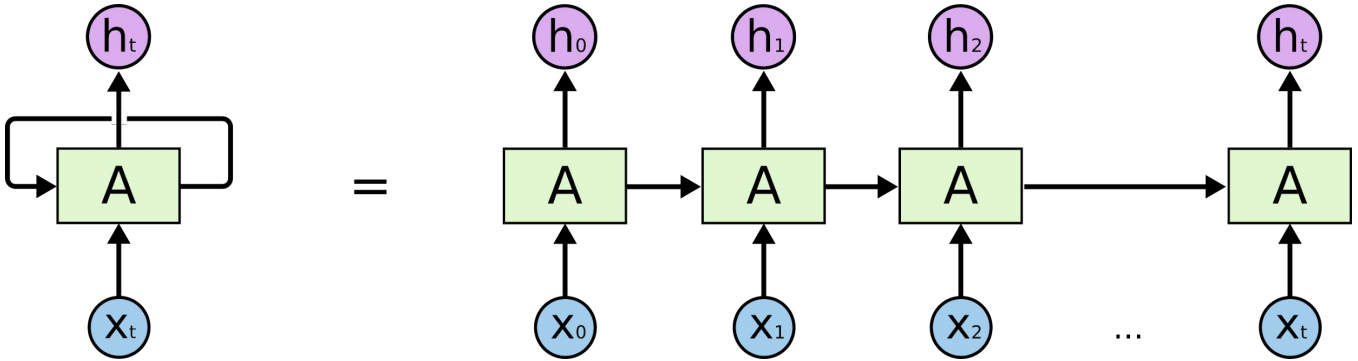
Sequence to sequence — transformation of input sequences into output sequences

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

Recap: RNN



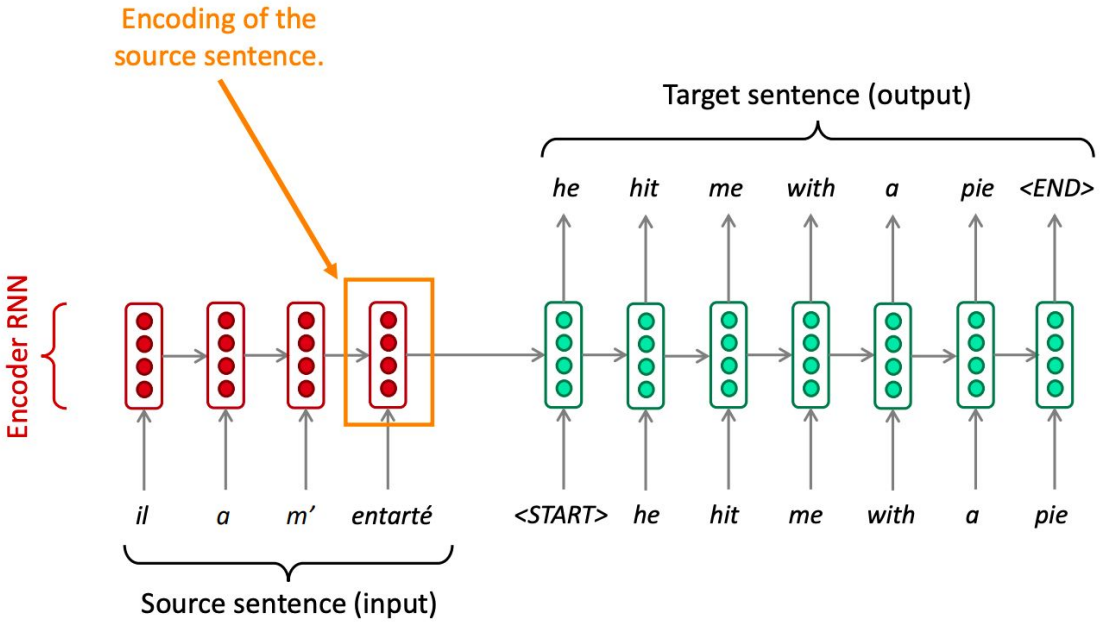
Recap: RNN



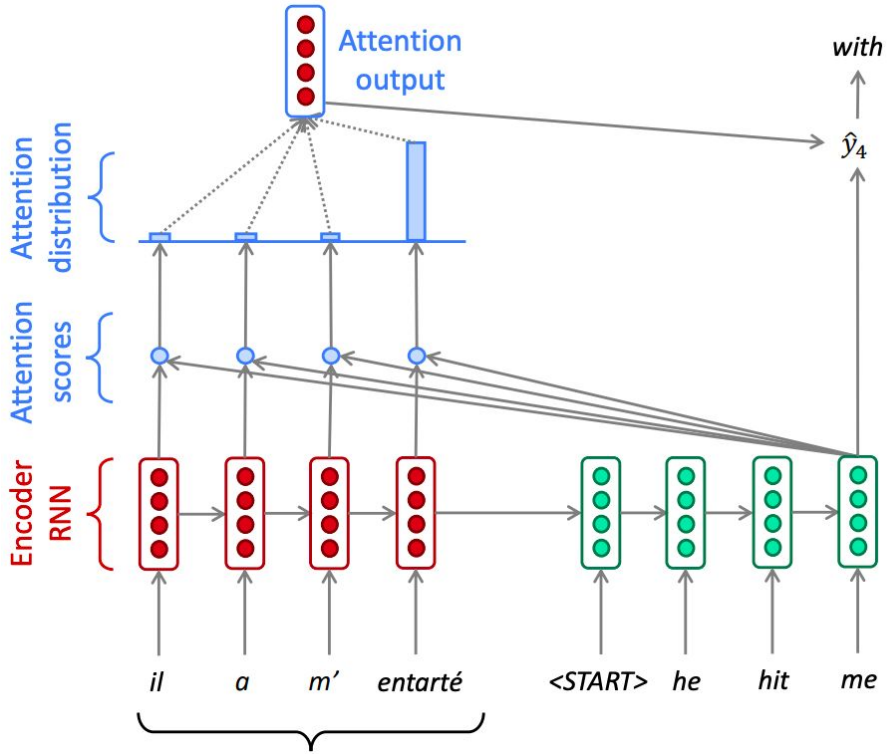
Problems:

- Learns slow
- Vanishing / exploding gradients
- Catastrophic forgetting

Recap: RNN


Encoding of the source sentence.
Target sentence (output)
he
hit
me
with
a
pie
<END>
Encoder RNN
il
a
m'
entarté
<START>
he
hit
me
with
a
pie
Source sentence (input)

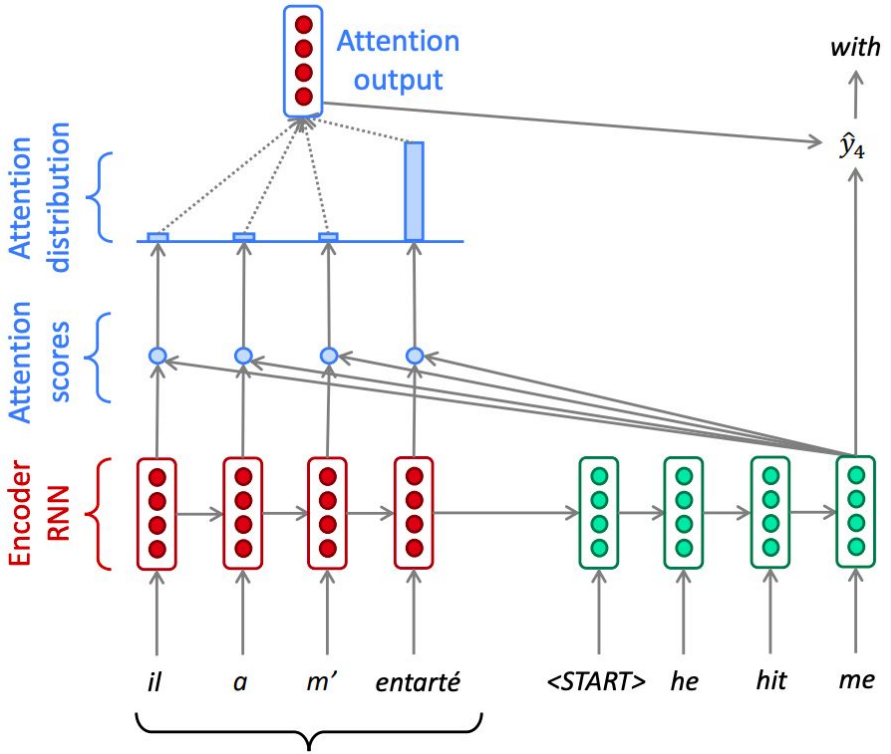
Recap: Attention



Recap: Attention

Attention:

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



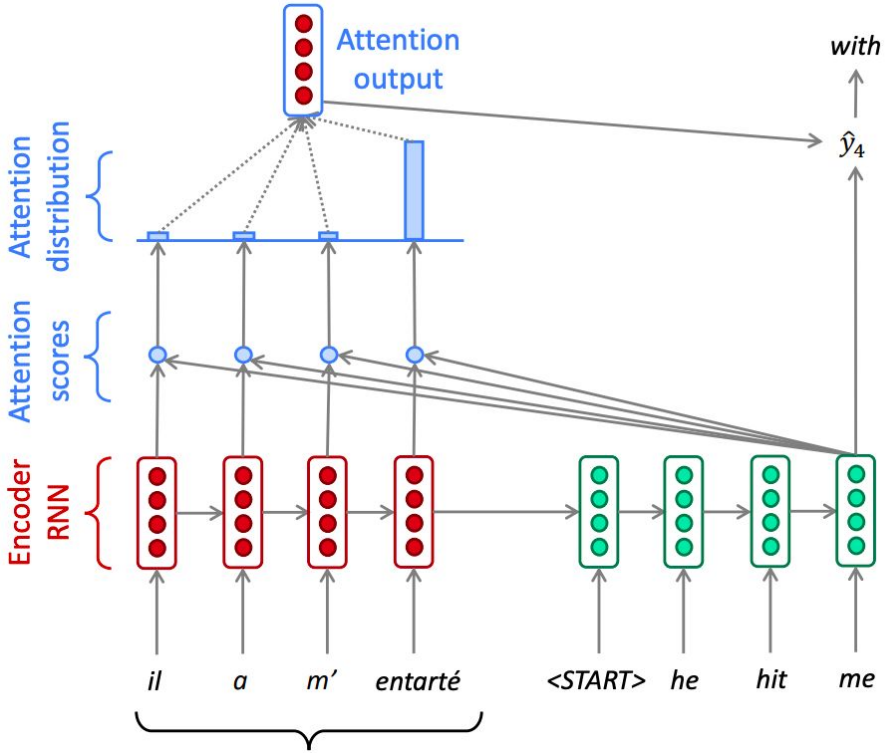
Recap: Attention

Attention:

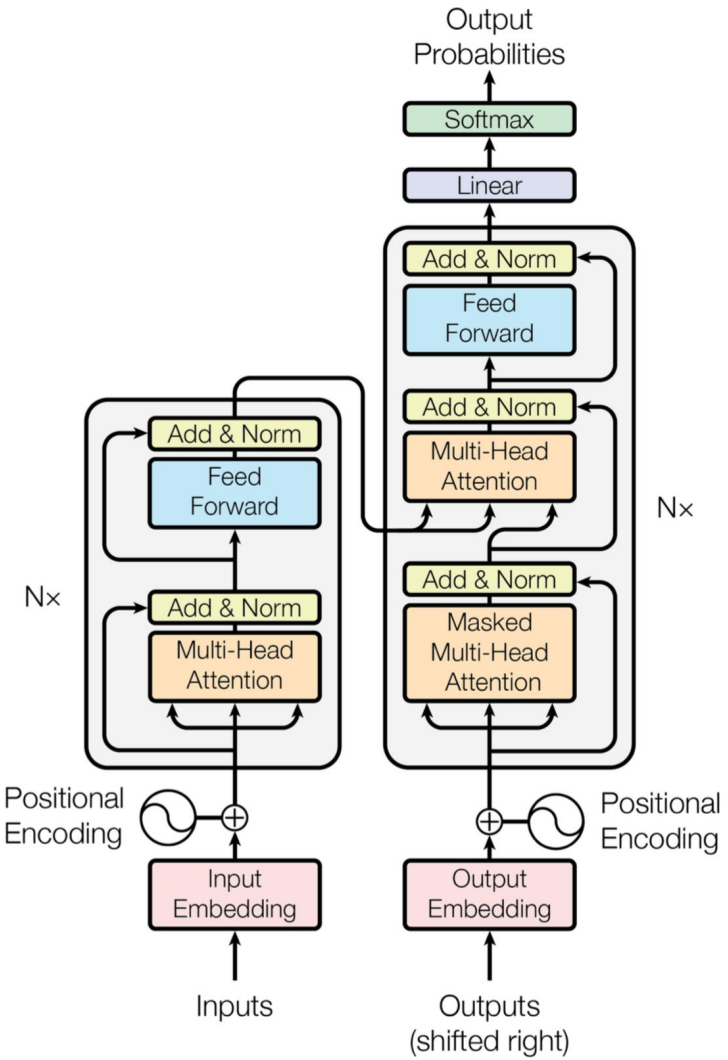
- Improves performance
- Helps with vanishing gradients
- Solves the bottleneck problem
- Helps with interpretability

BUT:

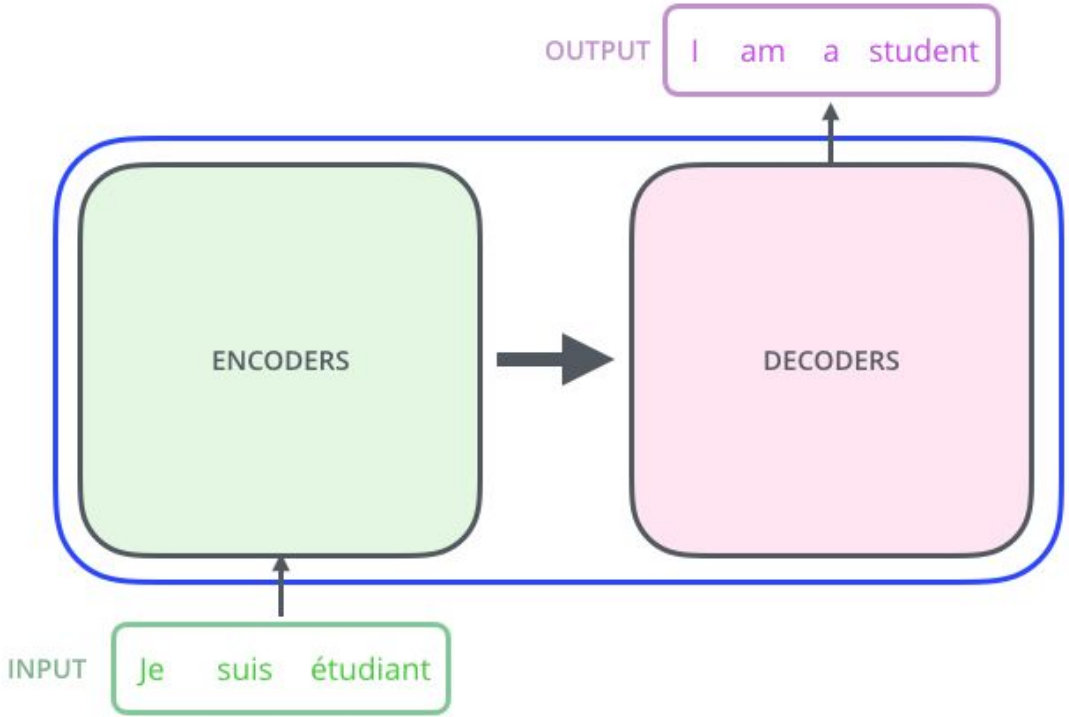
Models get more and more complex and the computations still can not be done in parallel therefore **SLOW**



Transformer



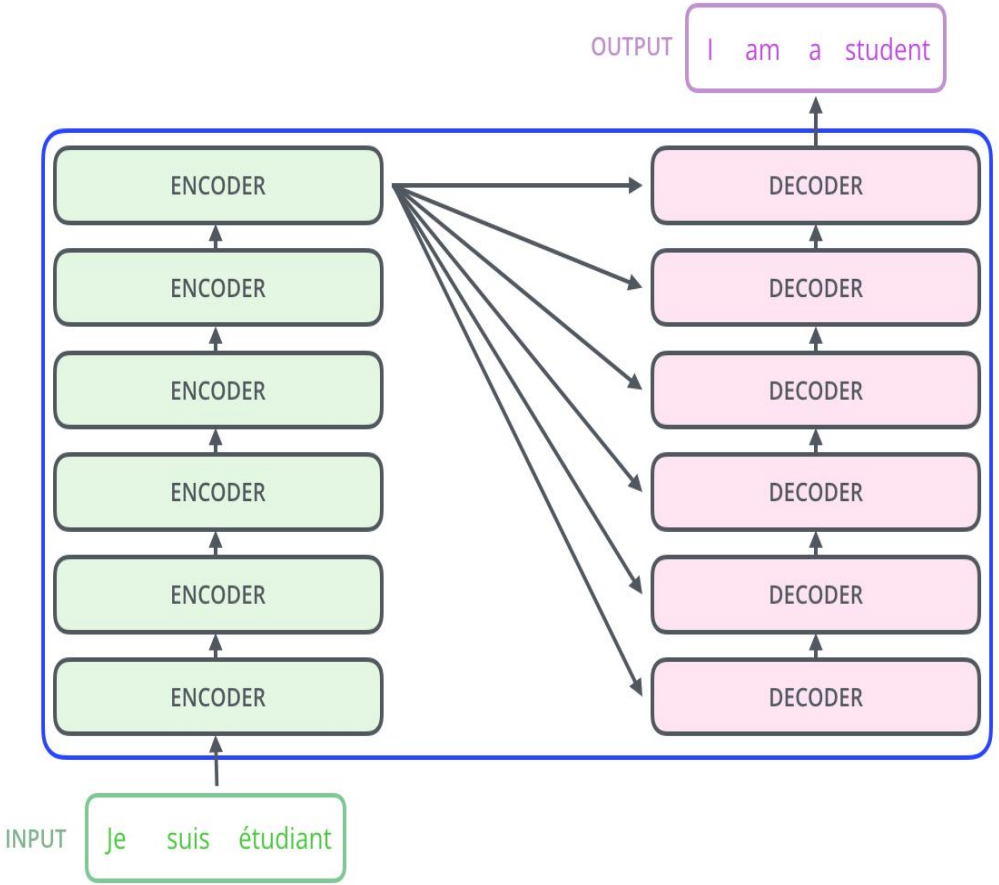
Architecture



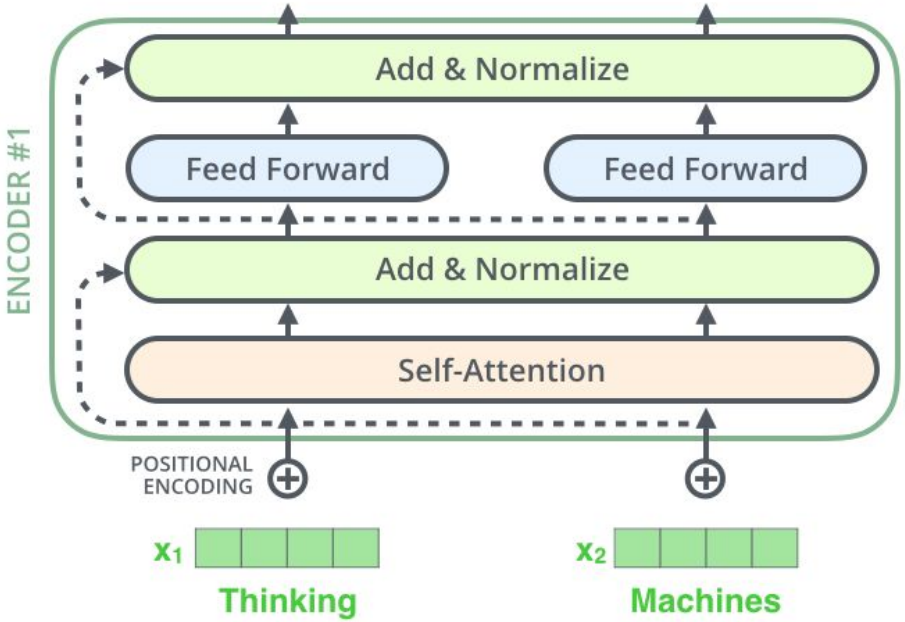
Architecture

Encoder and Decoder:

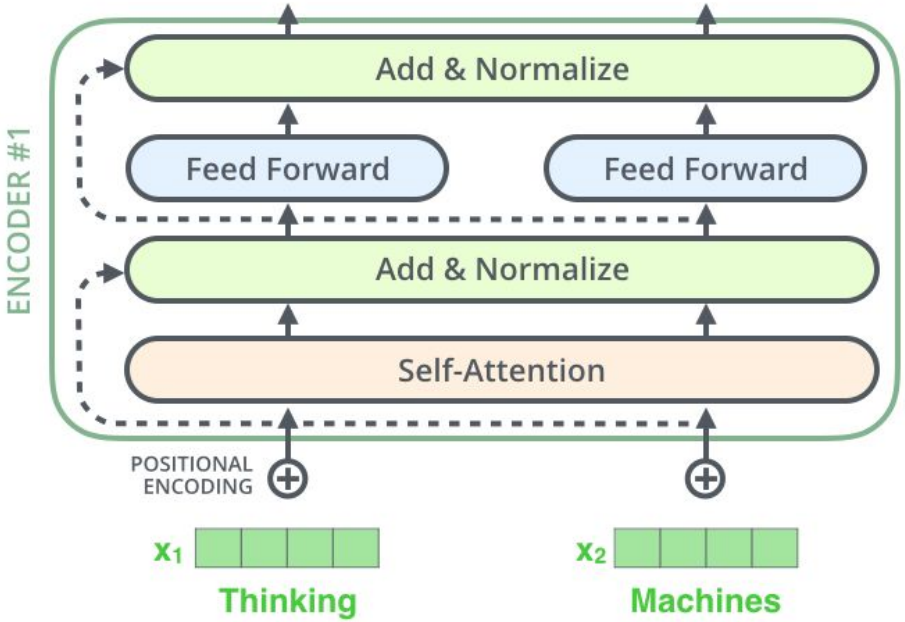
- Consist of blocks
- The blocks have the same architecture
- The blocks do not share weights

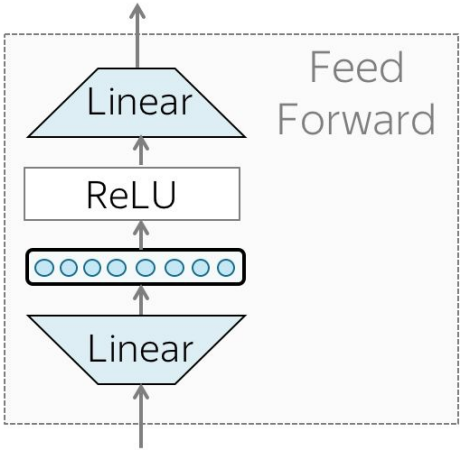


Encoder

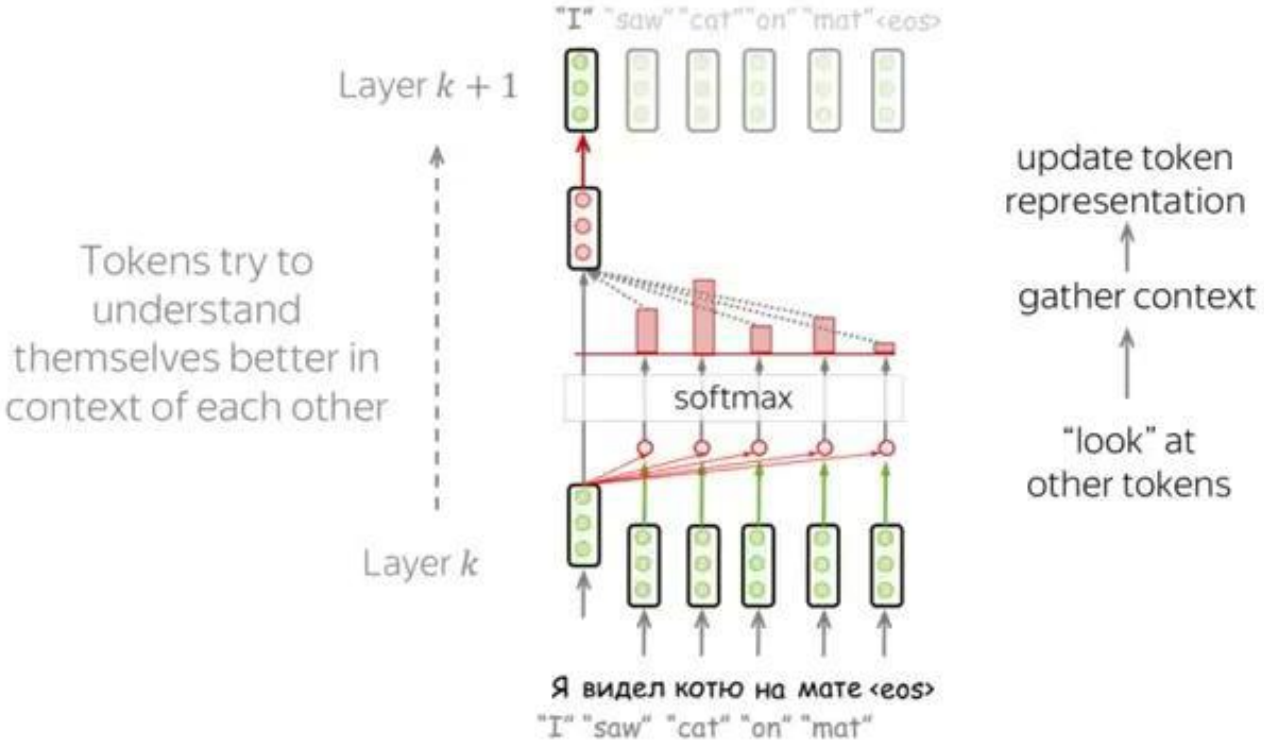

ENCODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Self-Attention
POSITIONAL ENCODING
x1
x2
Thinking
Machines

Encoder


ENCODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Self-Attention
POSITIONAL ENCODING
x1
x2
Thinking
Machines


Feed Forward
Linear
ReLU
Linear

Self-attention



Self-attention

Each vector receives three representations (“roles”)

$W_Q \times = $ Query: vector **from** which the attention is looking

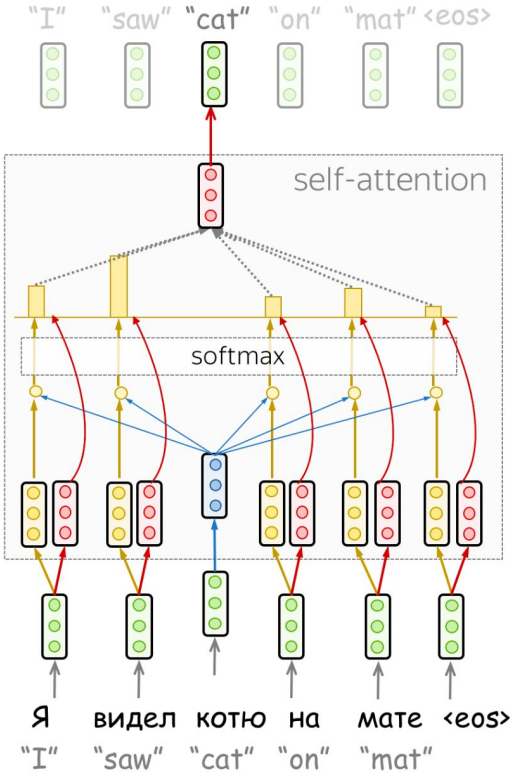
“Hey there, do you have this information?”

$W_K \times = $ Key: vector **at** which the query looks to compute weights

“Hi, I have this information – give me a large weight!”

$W_V \times = $ Value: their weighted sum is attention output

“Here’s the information I have!”



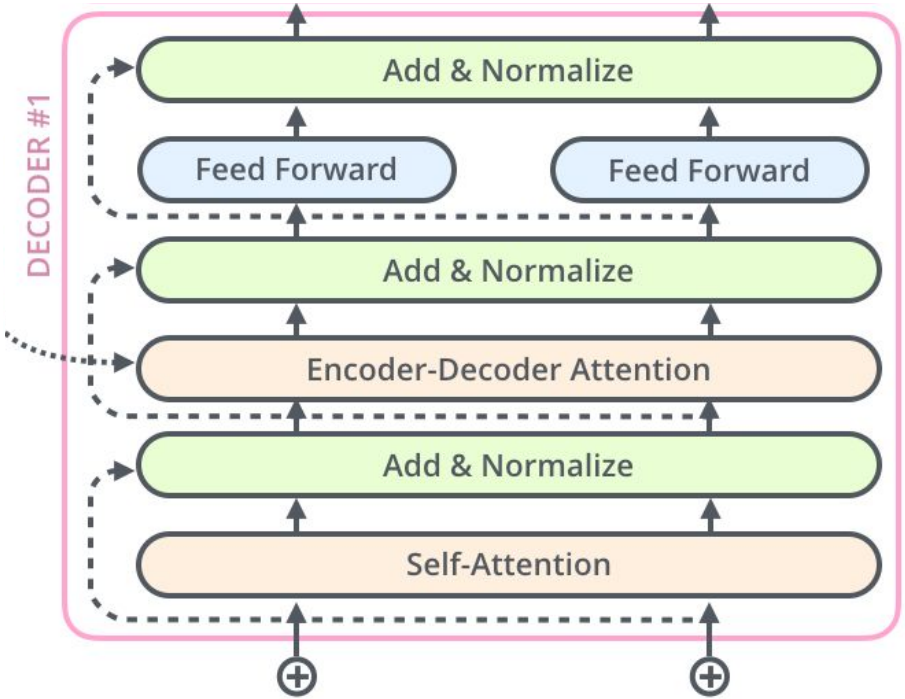
Self-attention

$$Attention(q, k, v) = \overbrace{softmax\left(\frac{qk^T}{\sqrt{d_k}}\right)v}^{\text{Attention weights}}$$

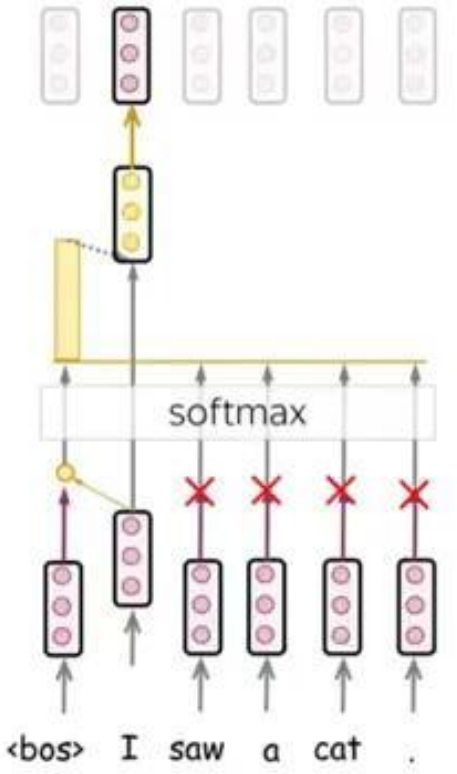
from (q) to (k, v)

vector dimensionality of K, V

Decoder


DECODER #1
Add & Normalize
Feed Forward
Feed Forward
Add & Normalize
Encoder-Decoder Attention
Add & Normalize
Self-Attention

Decoder self-attention

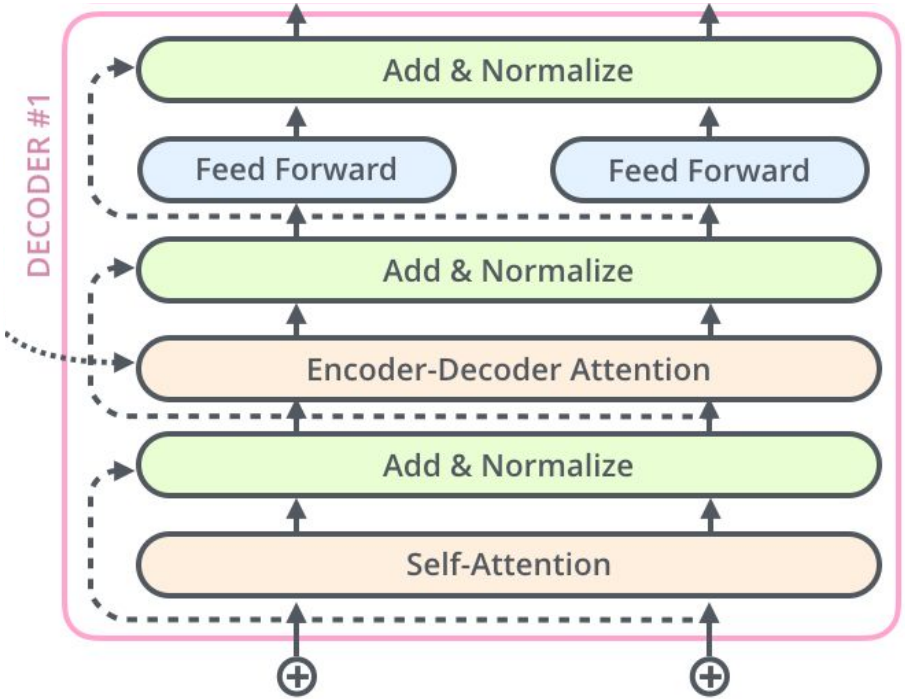


update token representation

gather context

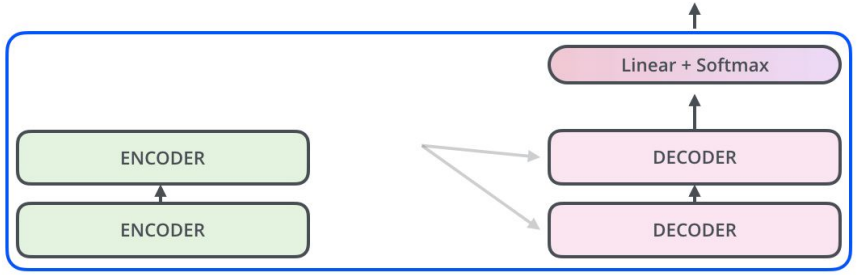
"look" at the previous tokens (future tokens are masked out)

Decoder


DECODER #1
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



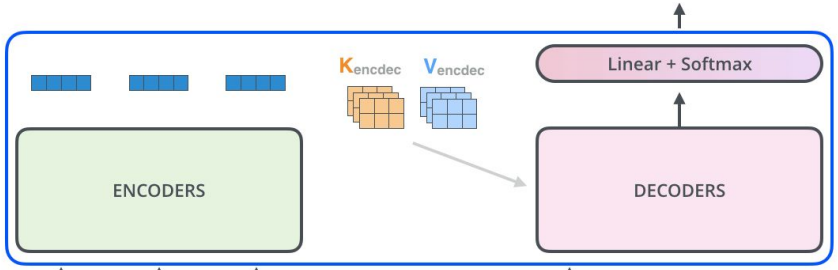
EMBEDDING WITH TIME SIGNAL

EMBEDDINGS

INPUT Je suis étudiant

Decoding time step: 1 (2) 3 4 5 6

OUTPUT I



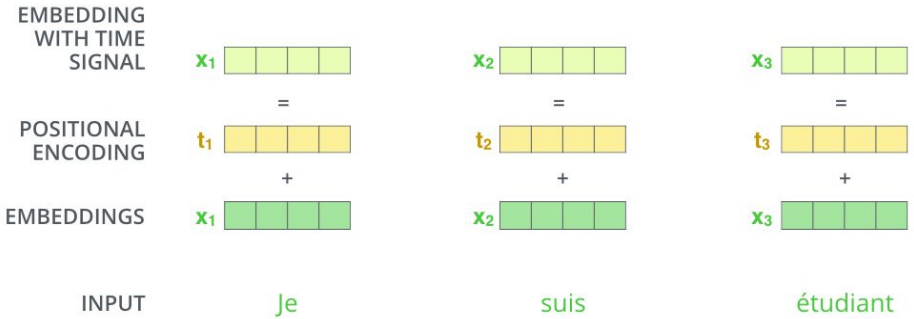
EMBEDDING WITH TIME SIGNAL

EMBEDDINGS

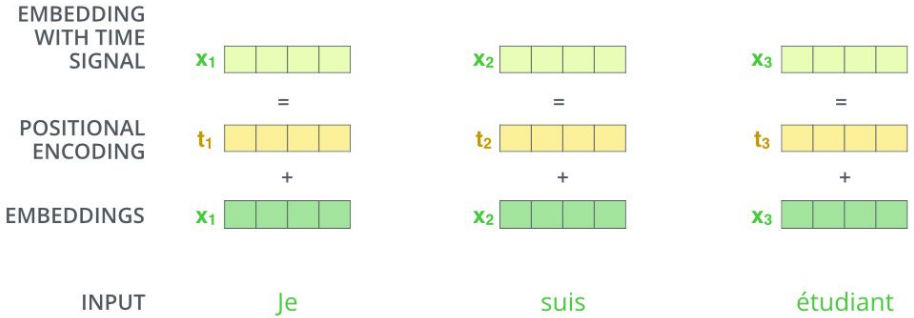
INPUT Je suis étudiant

PREVIOUS OUTPUTS I

Positional information



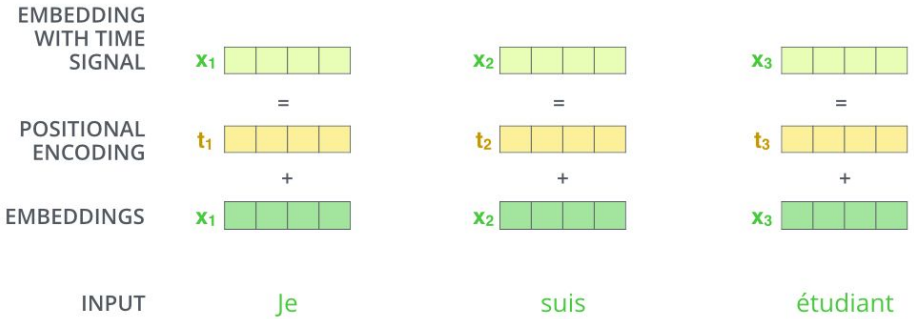
Positional information



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

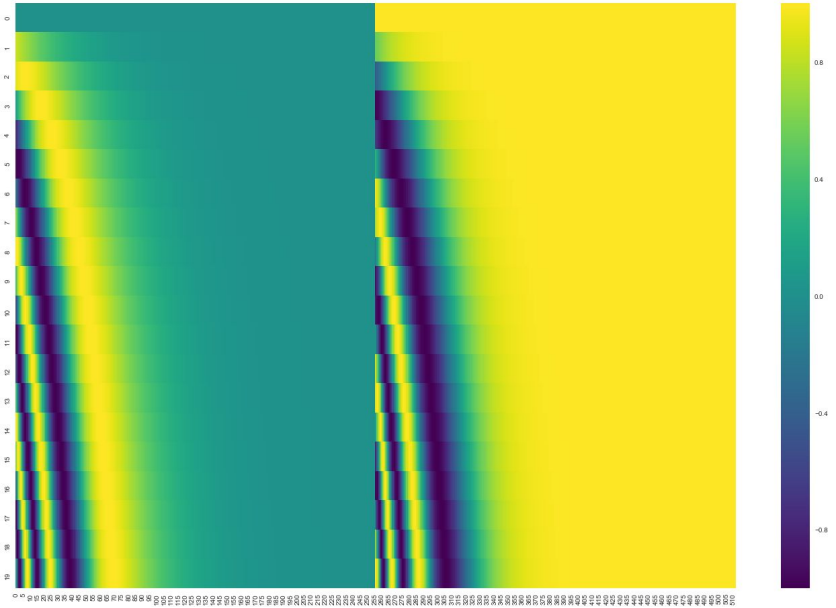
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Positional information

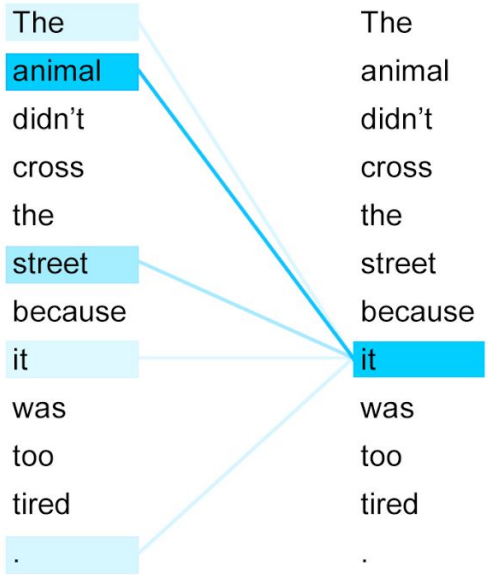


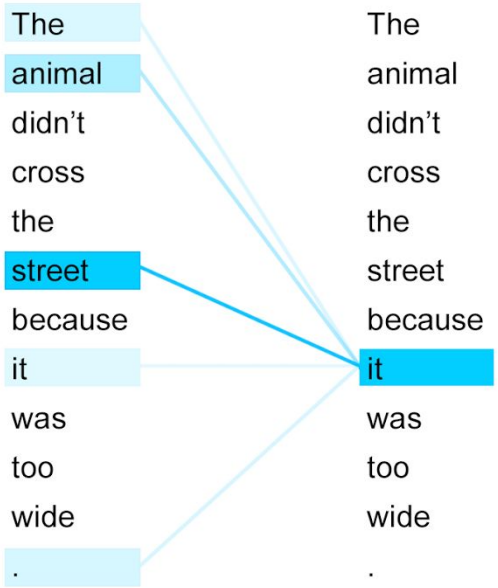
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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



Self-attention illustration





Вопросы

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

Resources

- https://arxiv.org/pdf/1706.03762.pdf (original paper)
- https://lena-voita.github.io/nlp_course/seq2seq_and_attention.html (blog post / NLP textbook)
- http://jalammar.github.io/illustrated-transformer/ (classic detailed explanation)
- 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 (paper blog post)