Katya Artemova

# Transformers

Sequence to sequence models

2021

# About myself

- PostDoc at HSE University
- Research and teach Natural Language Processing (NLP)
- My main research areas are:
  - Dialog systems
  - Interpretation of deep neural networks
  - NLP for Digital Humanities

# Transformers

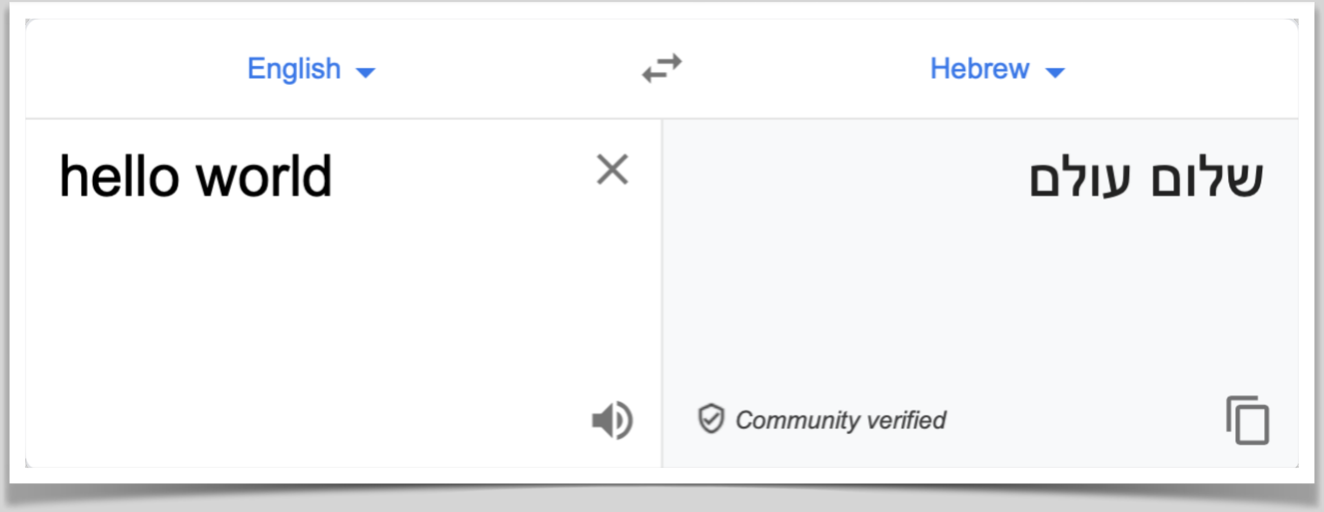
## Sequence to sequence models

- Intro to seq2seq models
- Transformer
    - Inside Encoder blocks
    - Inside Decoder blocks
- Random facts about machine translation
- Takeaways

# Intro to seq2seq models

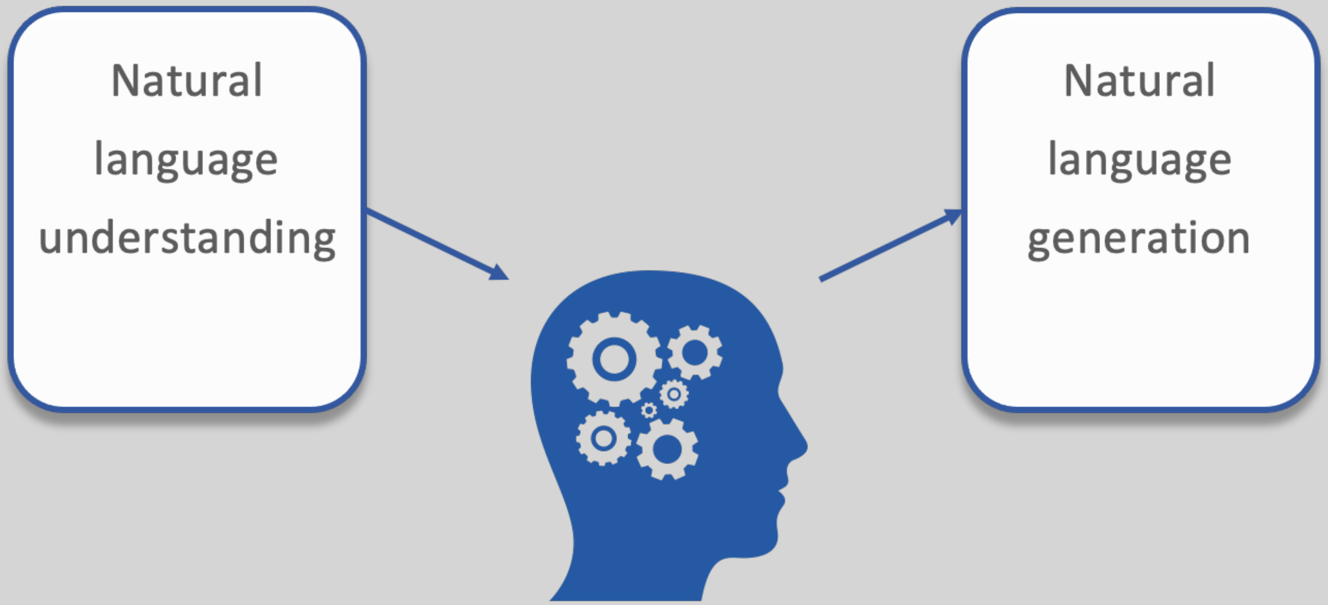
# Machine translation

## We use MT on a daily basis…



# Sequence to sequence models

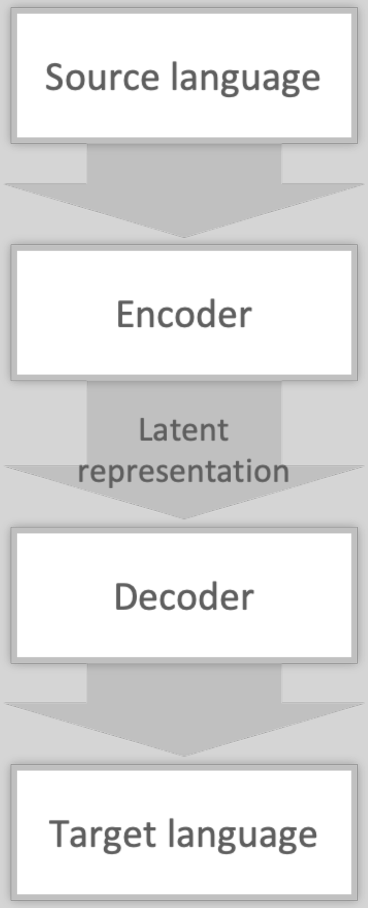
## Translate a sentence from one language into another



# Sequence to sequence models

## Encoder-decoder models

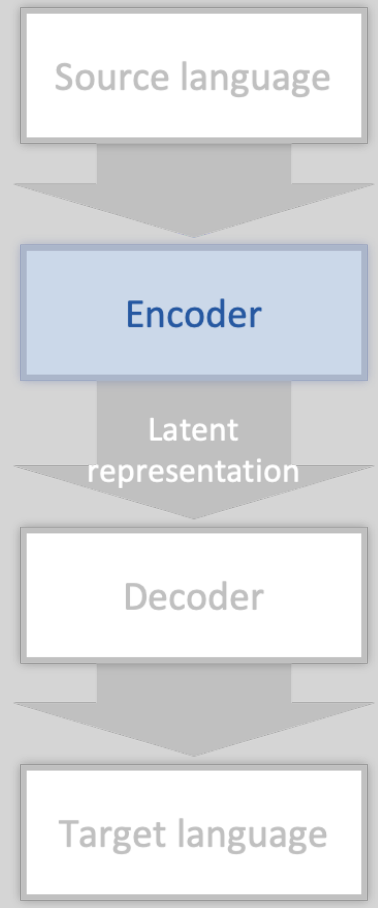
- Seq2seq models consist of two parts



# Sequence to sequence models

## Encoder-decoder models

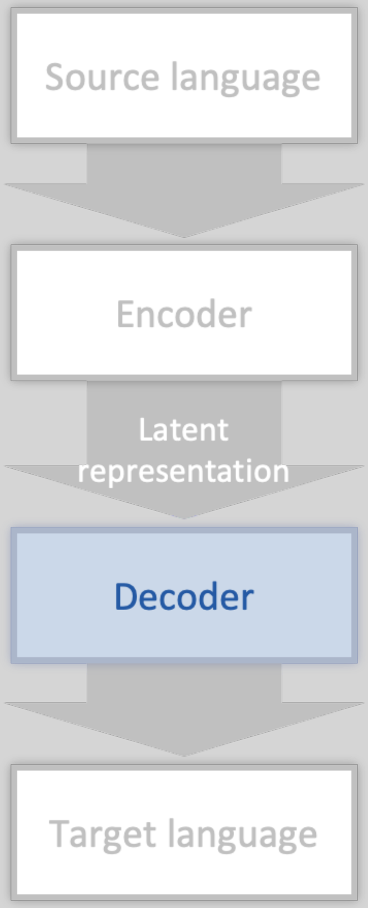
- Seq2seq models consist of two parts
- The encoder inputs a sentence in source language



# Sequence to sequence models

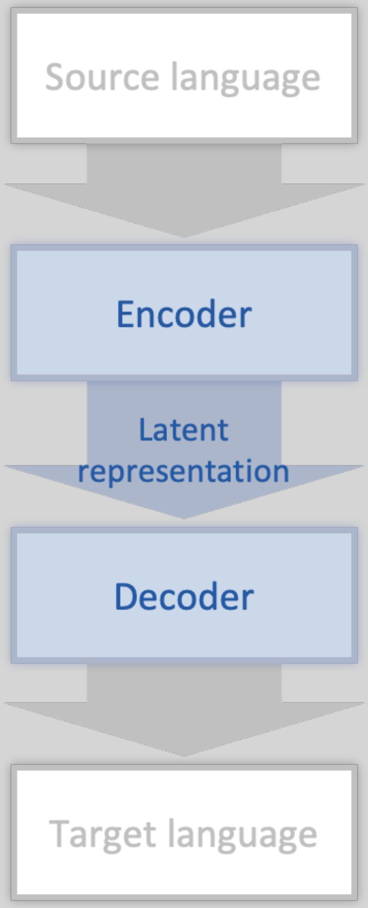
## Encoder-decoder models

- Seq2seq models consist of two parts
- The encoder inputs a sentence in the source language
- The decoder outputs a sentence in the target language



# Sequence to sequence models
## Encoder-decoder models

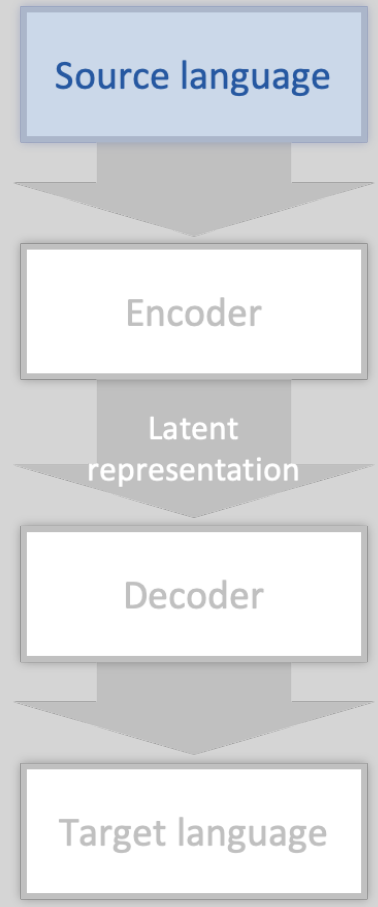
- Seq2seq models consist of two parts
- The encoder inputs a sentence in the source language
- The decoder outputs a sentence in the target language
- The latent representation is a vector representation of the input sentence



# Machine translation

## Problem formulation

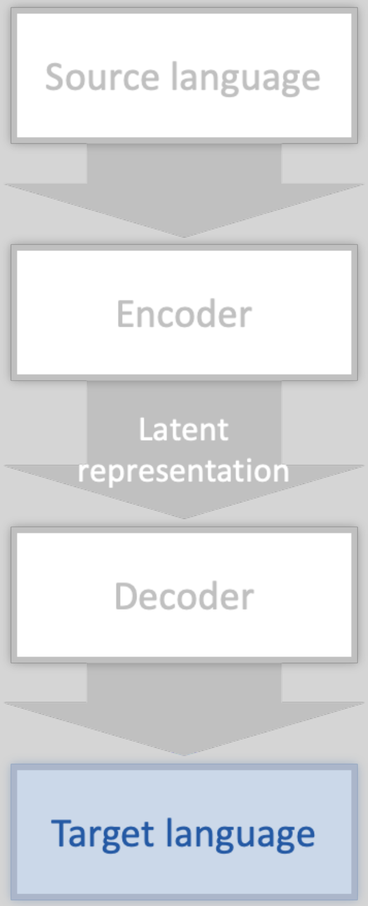
- Source sentence:
  $\mathbf{x}_{source} = (x_1, \ldots, x_n), x_i \in V_{source}$



# Machine translation

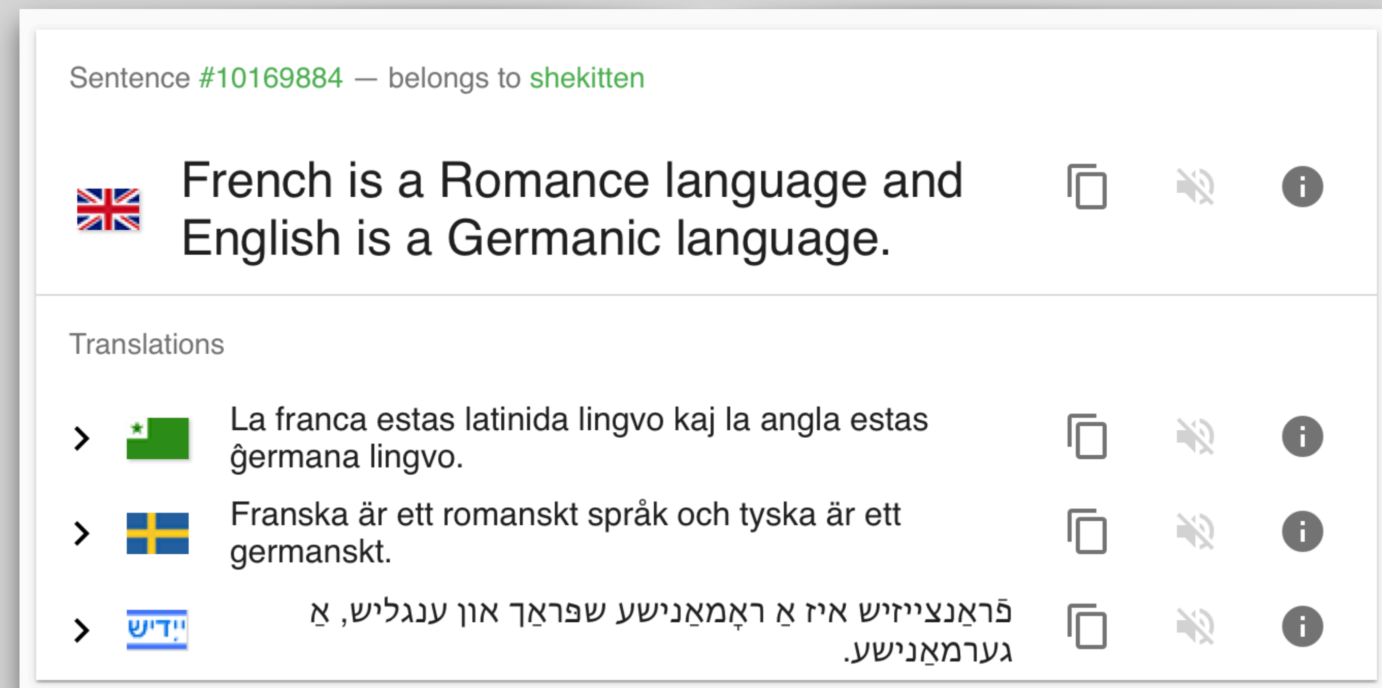
## Problem formulation

- Source sentence:
  $\mathbf{x}_{source} = (x_1, \ldots, x_n), x_i \in V_{source}$
- Target sentence:
  $\mathbf{y}_{target} = (y_1, \ldots, y_m), y_i \in V_{target}$



# Machine translation data

## Parallel corpus

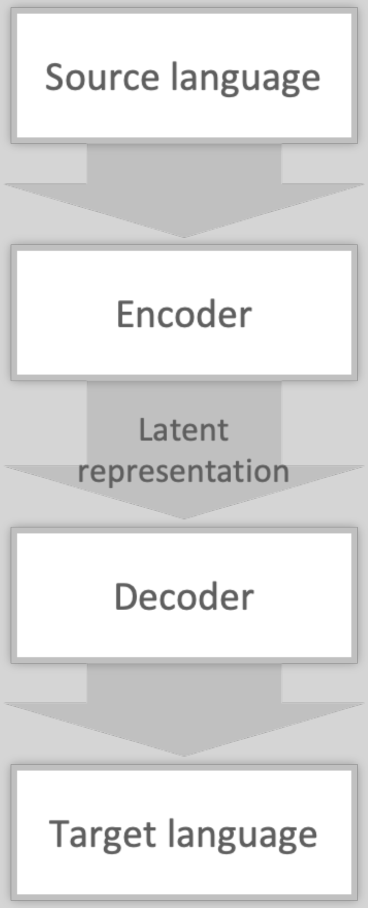



https://tatoeba.org/

# Machine translation

## Problem formulation

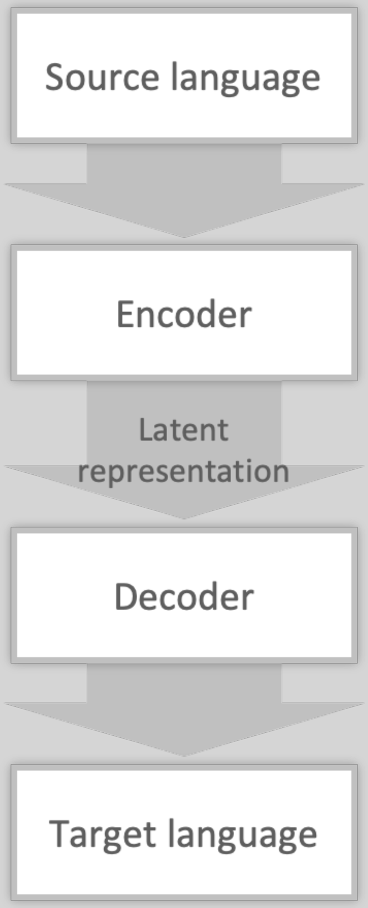
- Source sentence: $\mathbf{x}_{source} = (x_1, \ldots, x_n), x_i \in V_{source}$
- Target sentence: $\mathbf{y}_{target} = (y_1, \ldots, y_m), y_i \in V_{target}$
- Our goal is to maximize $p(\mathbf{y} \mid \mathbf{x})$



# Machine translation

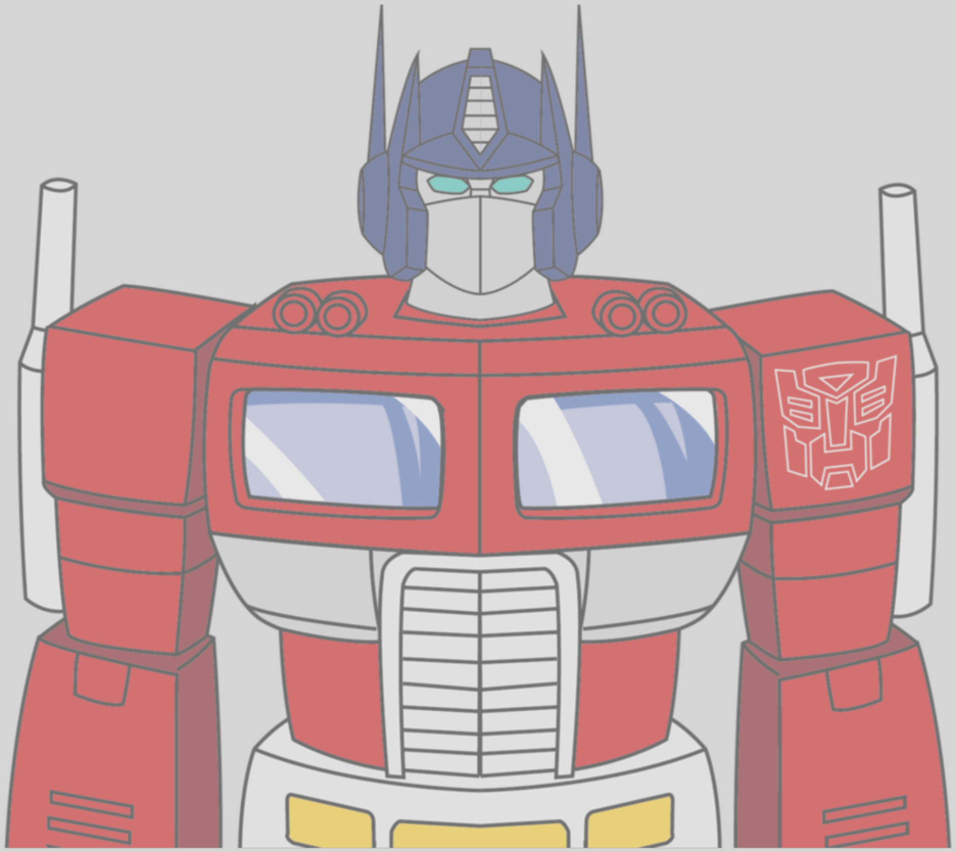
## Problem formulation

- Source sentence: $\mathbf{x}_{source} = (x_1, \ldots, x_n), x_i \in V_{source}$
- Target sentence: $\mathbf{y}_{target} = (y_1, \ldots, y_m), y_i \in V_{target}$
- Our goal is to maximize $p(\mathbf{y} \mid \mathbf{x})$




Objective function: $\mathcal{L}_\theta = \sum_{x,y \in C} \log p(\mathbf{y} \mid \mathbf{x}; \theta)$

# Transformer

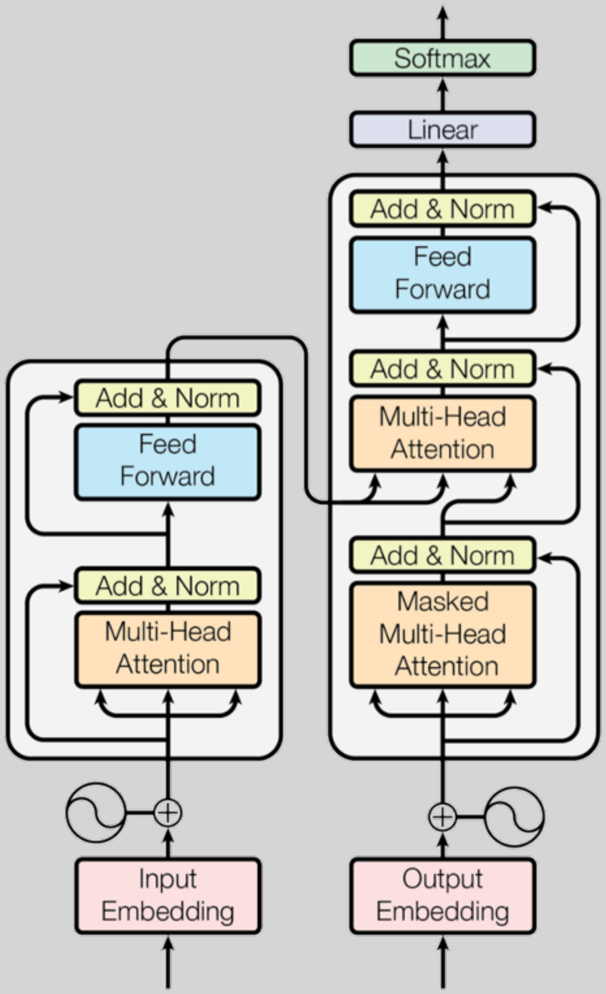


*Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In Advances in neural information processing systems, pp. 5998-6008. 2017.*

# Transformer

## Encoder-decoder model

- A faster and more efficient model for machine translation that the previous ones
- Core ideas:
  - Multi-head attention mechanism
  - Two stacks of layers






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

# Transformer

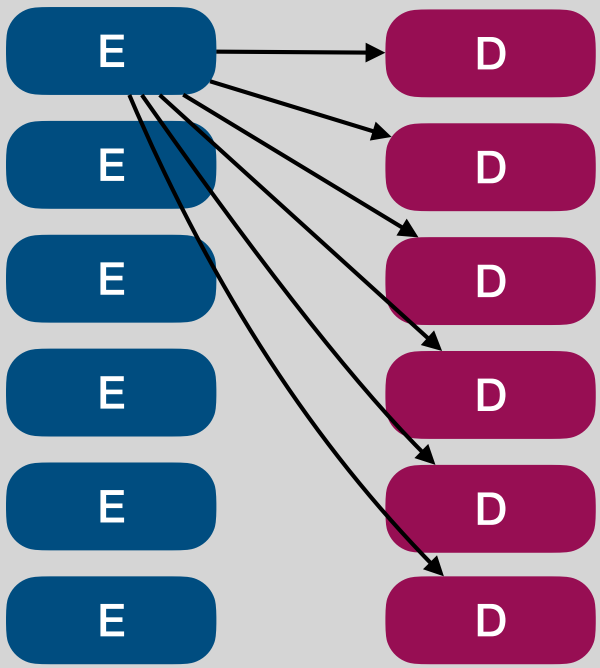
## Encoder-decoder model

- A faster and more efficient model for machine translation that the previous ones

- Core ideas:

  - Multi-head attention mechanism

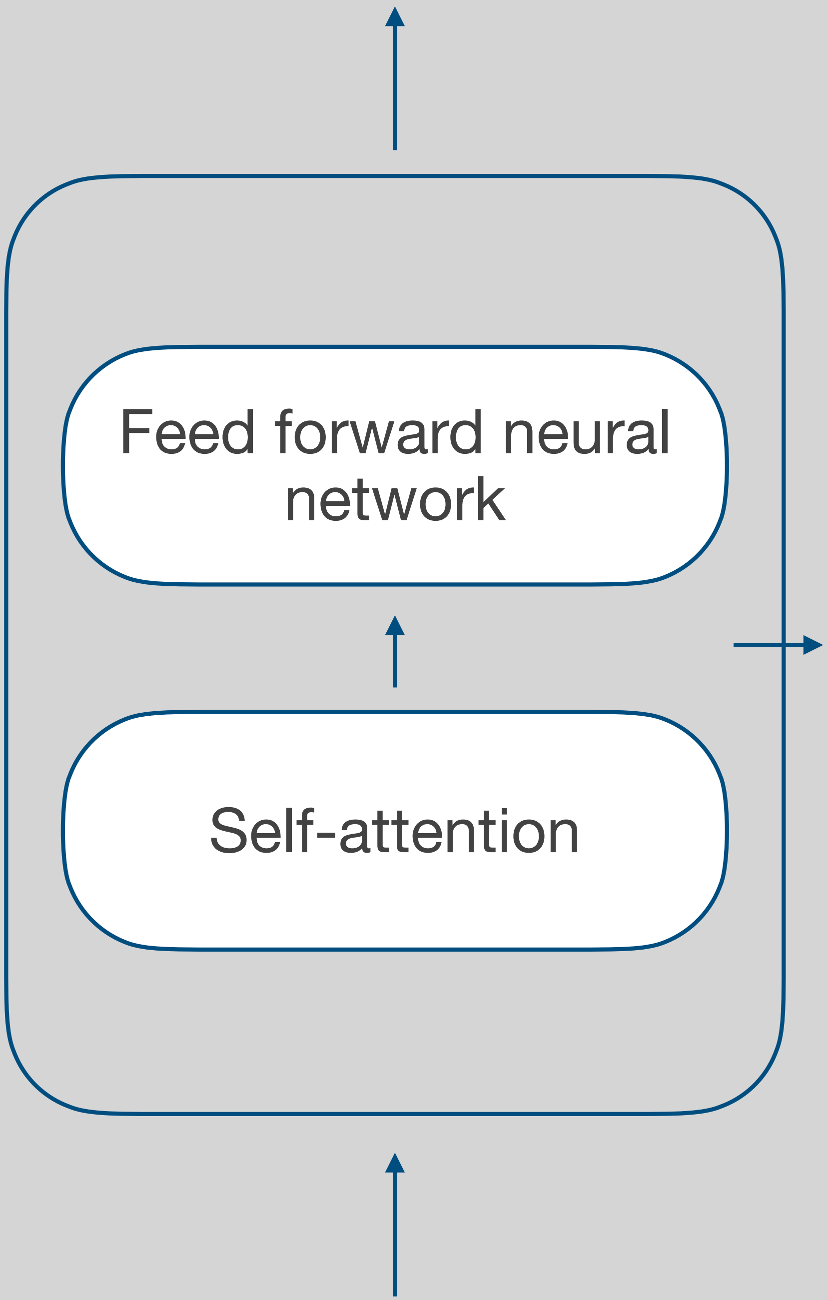
  - Two stacks of layers



# Transformer

## Encoder blocks

- The encoder block has two sublayers:
  - The self-attention layer helps to discover relations between words within sentence
  - The FFN layer aggregates outputs of the self-attention layer

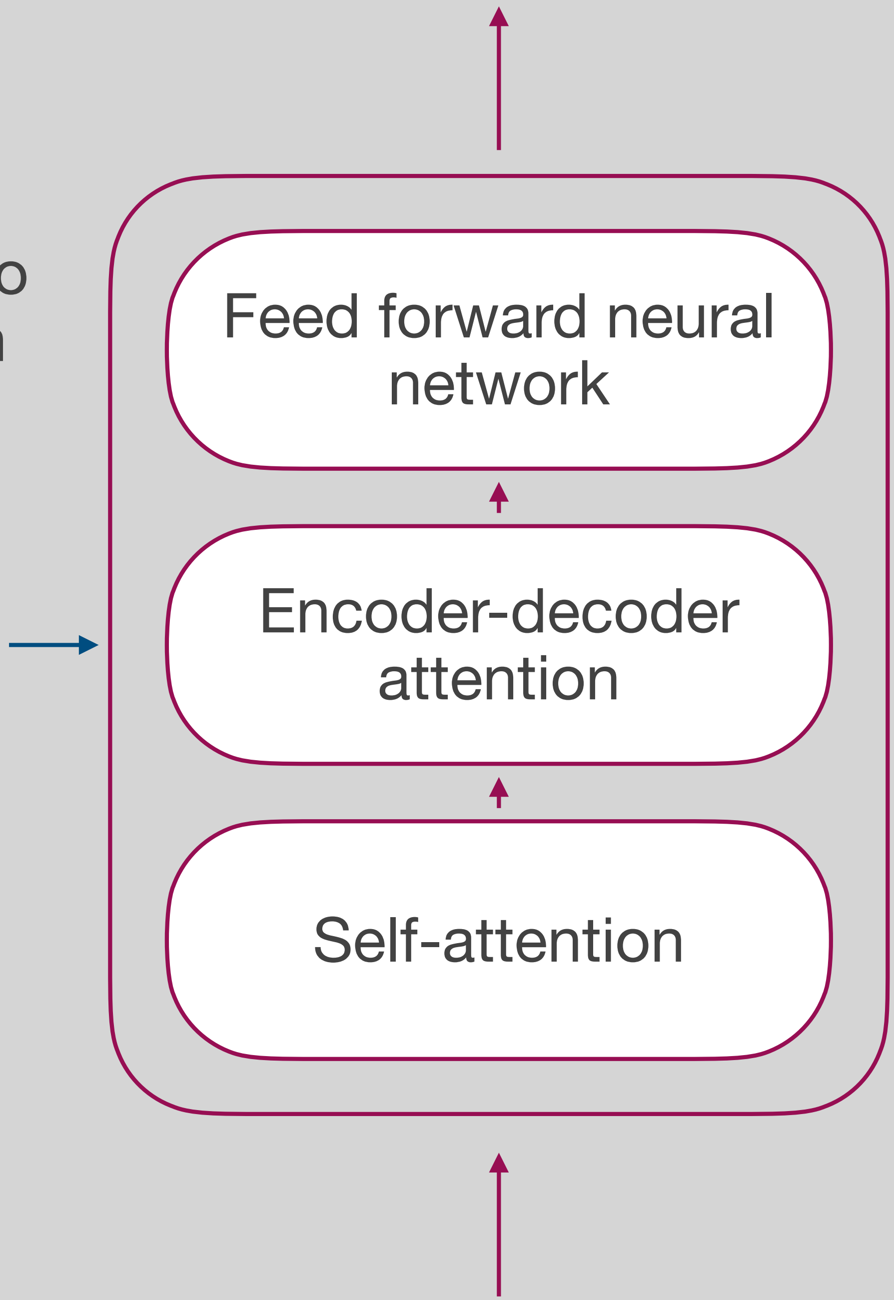


# Transformer

## Decoder blocks

- The decoder block has the same two layers, but between them there is an encoder-decoder attention layer

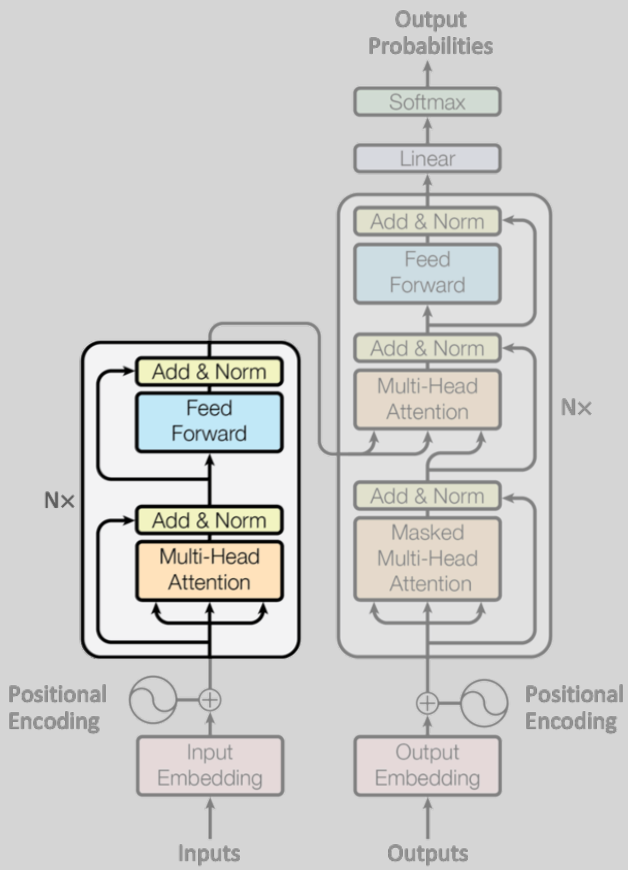
- An encoder-decoder attention layer helps the decoder to focus on different input words



# Transformer

## Encoder blocks

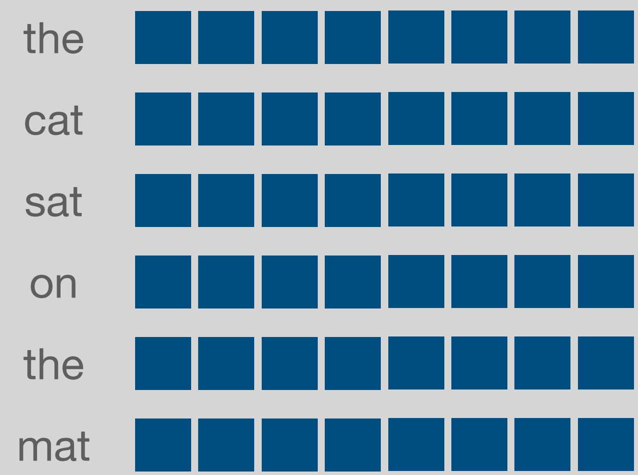
- The encoder consists of $N$ identical blocks
- Each block inputs the outputs of the previous block except the first one
- The first block inputs word embeddings



# Word embeddings

## Each word is a vector!

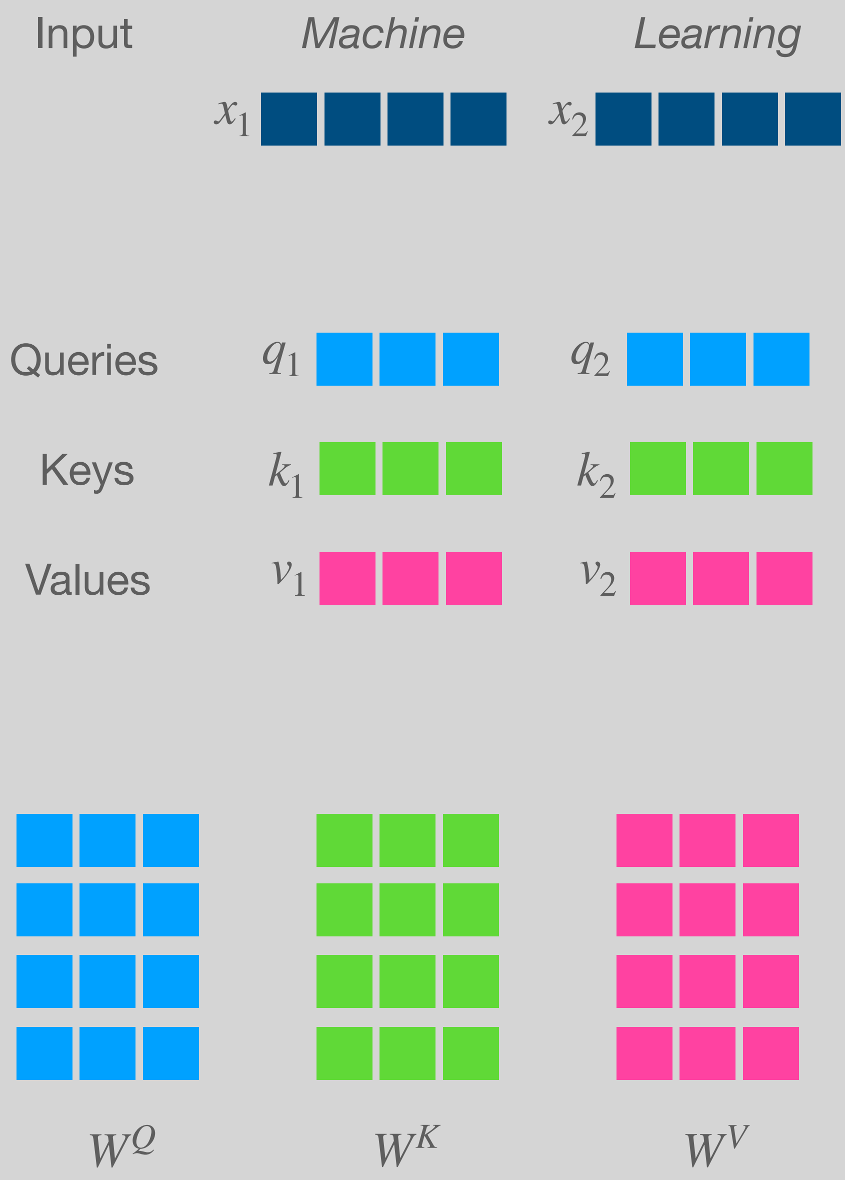
- Each word is represented with a single vector of size 512
- These vectors are dense
- The more similar words are, the closer their embeddings are $cos(x_i, x_j)$
- The embeddings are trainable, i.e. get updated when the whole model is trained
- The embeddings actually are the sum of word embeddings and positional encodings $x_i = e_i + pe_i$



# Inside the Encoder block

## Step 1. Self-attention

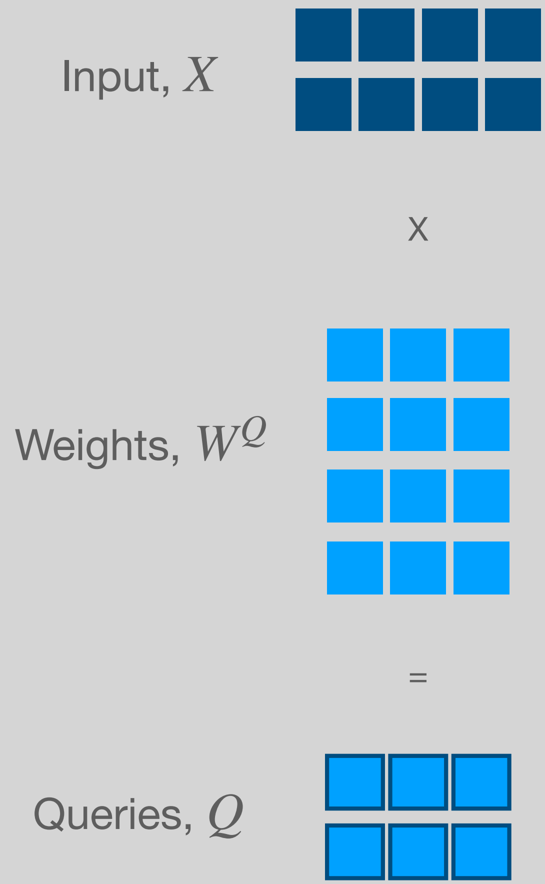
- Each word embedding is transformed into a query, a key and a value vectors
  - queries: $q_i = W^Q x_i$
  - keys: $k_i = W^K x_i$
  - values: $v_i = W^V x_i$
- Weight matrices $W^Q, W^K, W^V$ are trainable (=are updated during the training)



# Inside the Encoder block

## Step 1. Self-attention

- In matrix notation: word embeddings $X$ are multiplied by weight matrices:
  - queries: $Q = XW^Q$,
  - keys: $K = XW^K$
  - values: $V = XW^V$

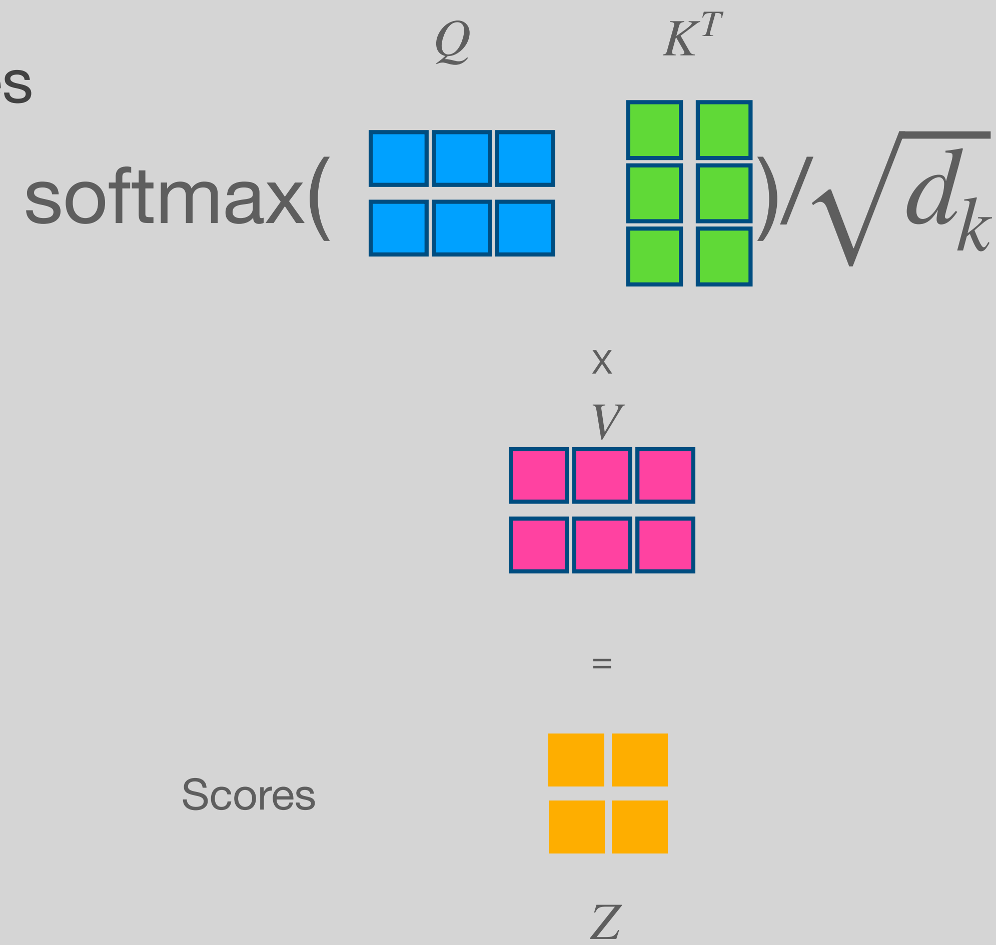


# Inside the Encoder block

## Step 1. Self-attention

- Score similarities between queries and keys: $\alpha_{11} = q_1 k_1$
- Scores are further divided by default value of 8 and fed into softmax. Finally, the normalised scores are multiplied by Value matrix.
- Using matrix notation:

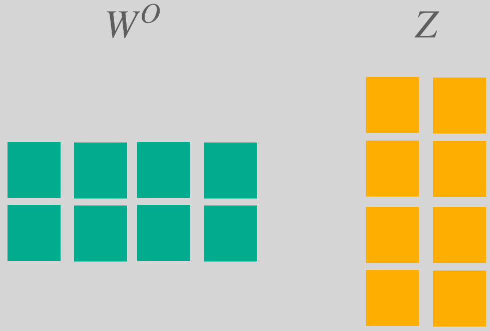
$$Z = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



# Inside the Encoder block

## Step 2. Multi-head self-attention

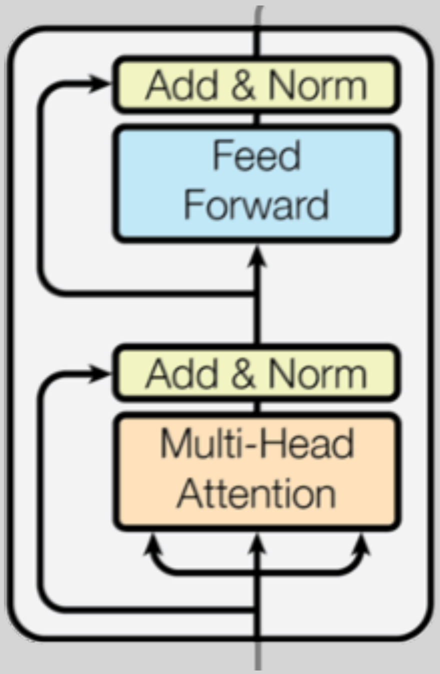
- Transformer used eight attention heads
- The outputs of eight heads are concatenated
  $Z = \text{concat}([Z^1, Z^2, \ldots, Z^8])$
- The output of the encoder block is $W^O Z$



# Inside the Encoder block

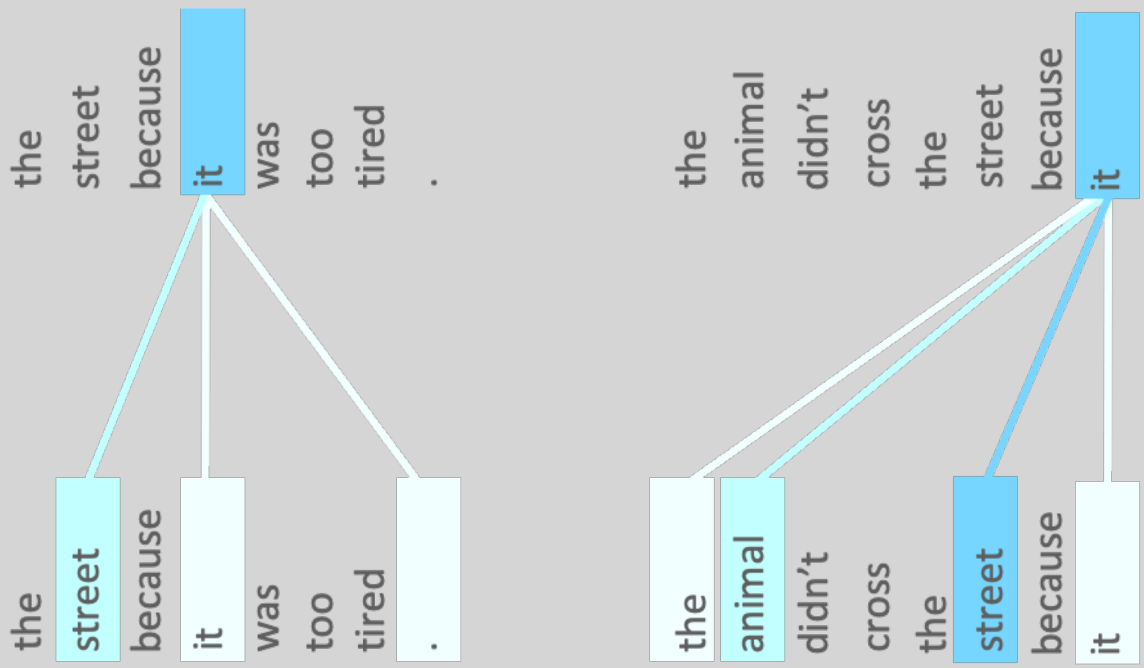
## Residual connections

- There are two Add & Norm layers insider the Encoder block
- Multi-head attention-mechanism inputs the matrix $X$ and outputs the matrix $Z$: $X \rightarrow \text{MHA} \rightarrow Z$
- The Add & Norm layers applies LayerNorm (subtract mean value and divide by standard deviation) to the sum of $X + Z$



# Inside the Encoder block

## Attention weights




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

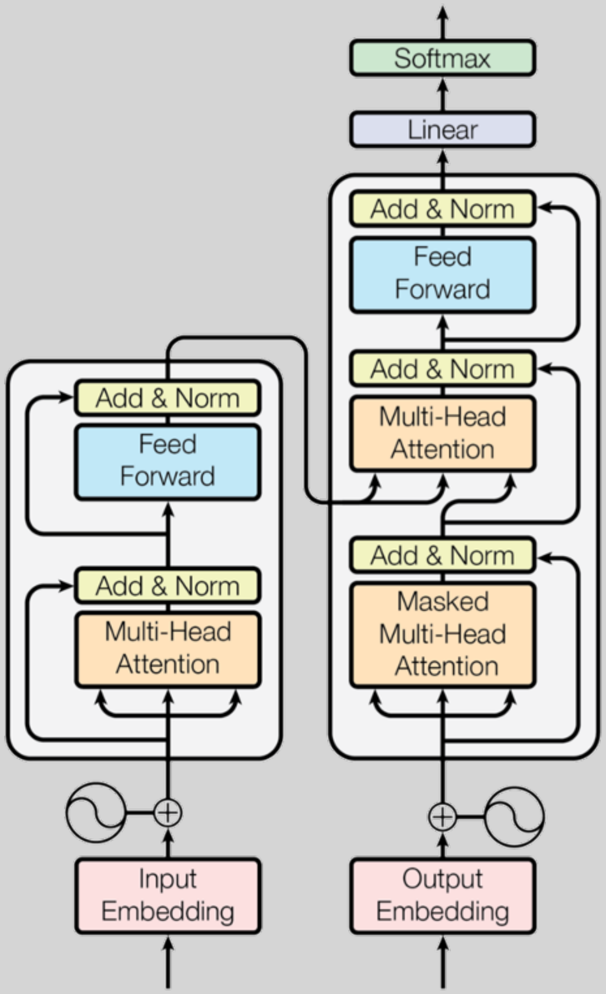
# Inside the Decoder block

## Encoder-decoder attention

- The encoder processes the input sequence

- The output of the top encoder block are key and value matrices $K_{encdec}, V_{encdec}$

- The decoder generates words one by one until the stop symbol is generated

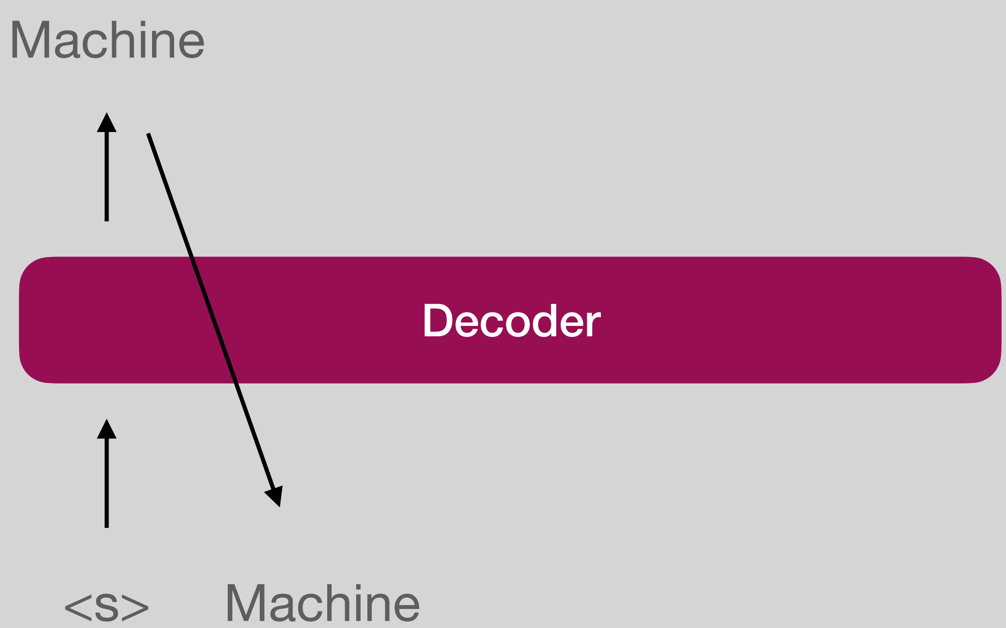


# Inside the Decoder block

## Word by word generation

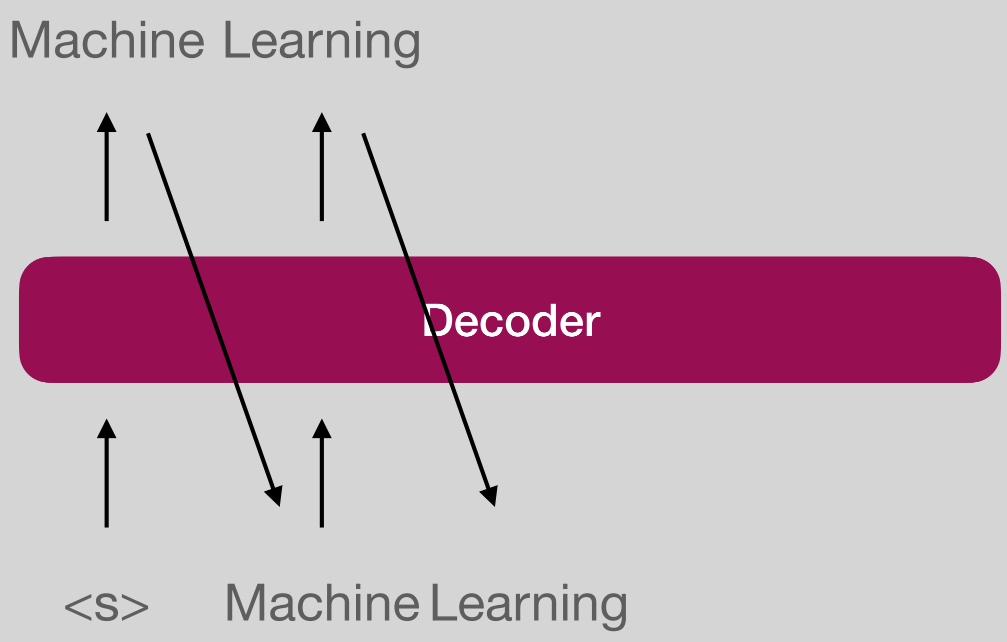
# Inside the Decoder block

## Word by word generation



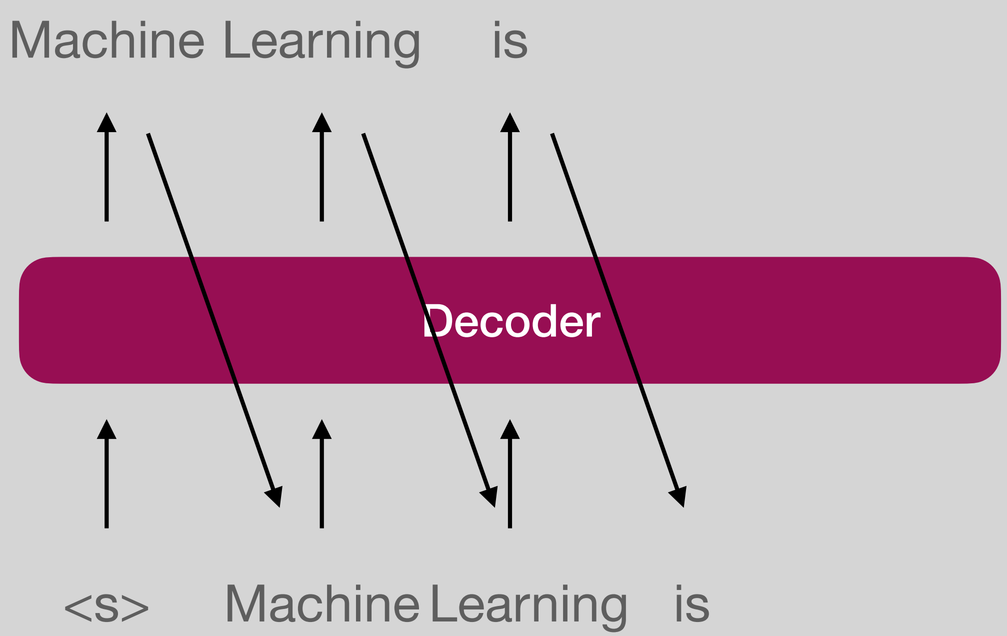
# Inside the Decoder block

## Word by word generation



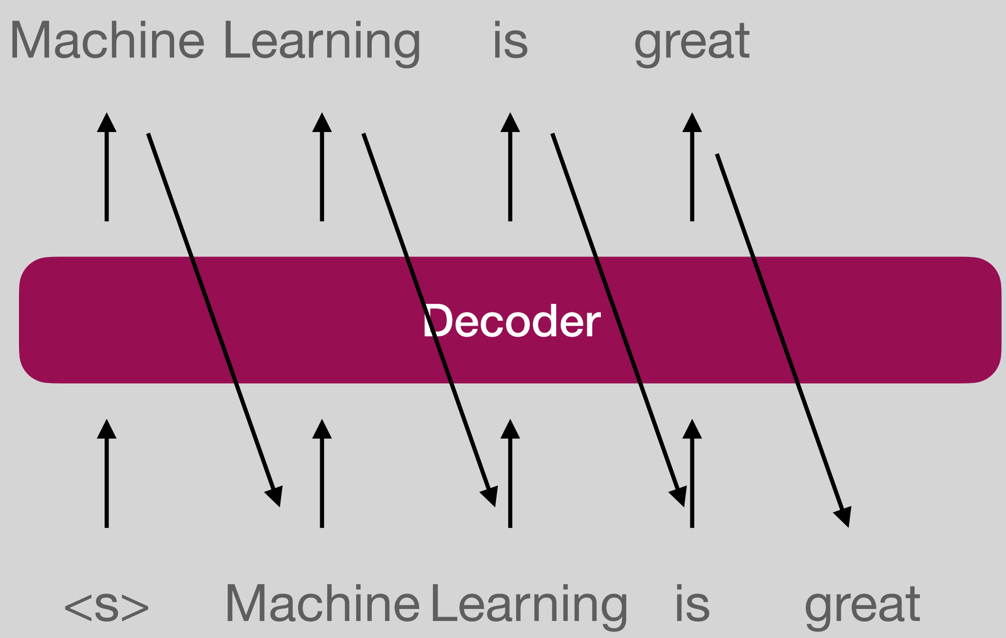
# Inside the Decoder block

## Word by word generation



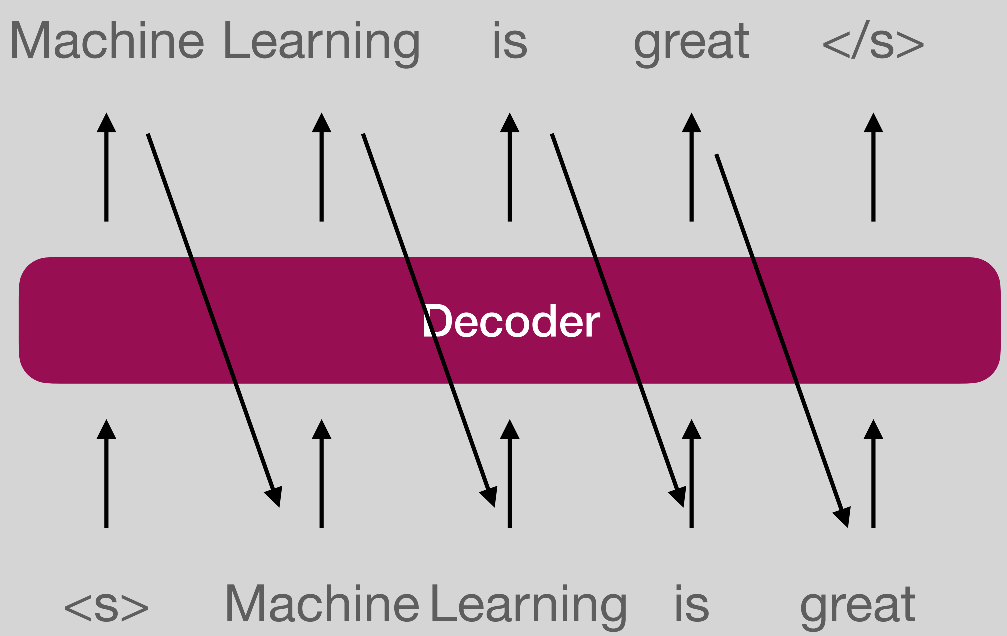
# Inside the Decoder block

## Word by word generation



# Inside the Decoder block

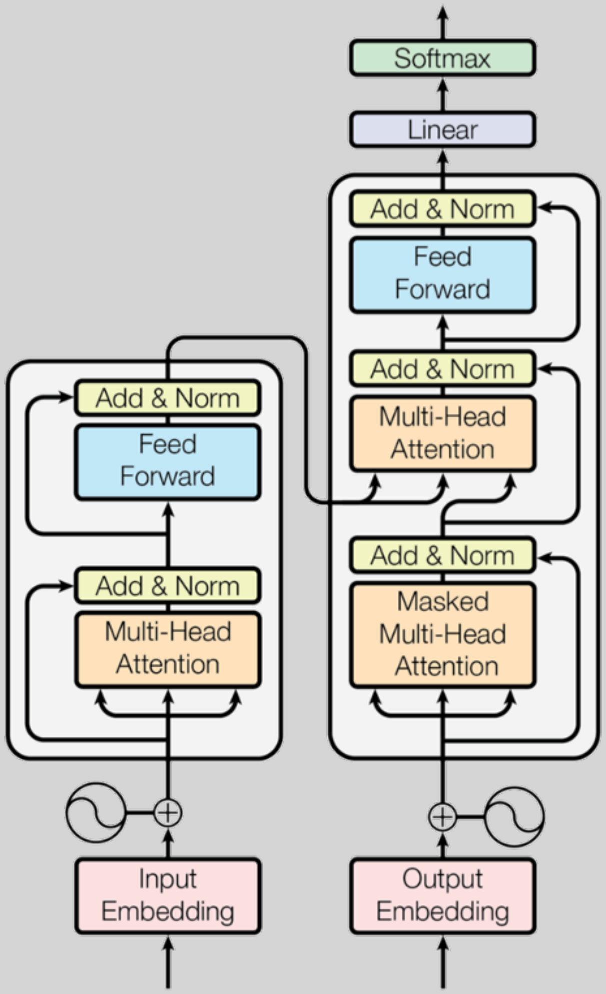
## Word by word generation



# Inside the Decoder block

## Encoder-decoder attention

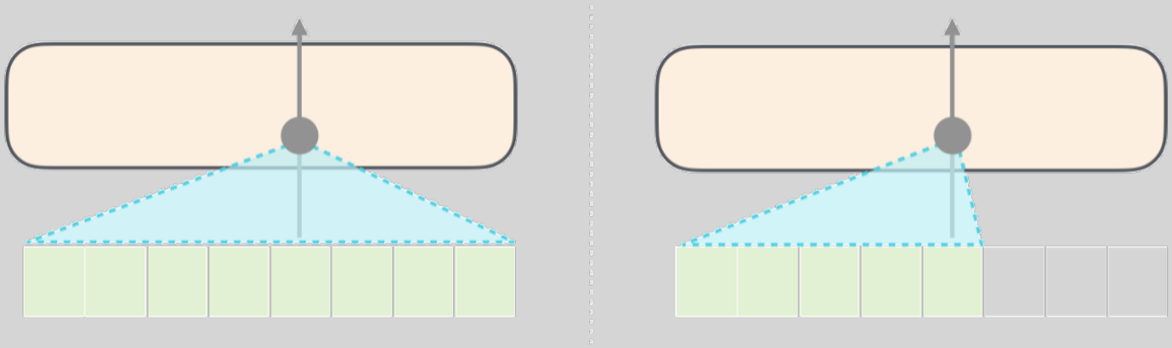
- The encoder processes the input sequence
- The output of the top encoder block are key and value matrices $K_{encdec}, V_{encdec}$
- The decoder generates words one by one until the stop symbol is generated
- The decoder transforms the input sequence $\mathbf{y}$ into the matrices $Q^{dec}, K^{dec}, V^{dev}$



# Inside the Decoder block

## Masked attention

- The attention mechanism in the Encoder is able to access the whole input sequence
- In the Decoder the masked attention mechanism is only allows to earlier words (to the left words)
- Future positions are masked by setting their weights to $-\infty$



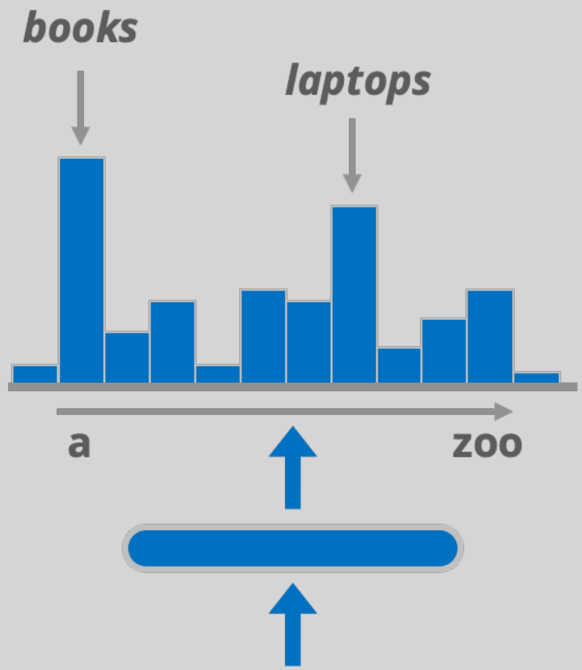
The Encoder's attention mechanism

The Decoder's attention mechanism

# Inside the Decoder block

## How to predict next word?

- The size of the output layer is equal to the size of target vocabulary: $|W^{O}_{dec}| = |V_{target}|$
- Each position corresponds to a single word
- The output scores are normalised by softmax, which turns the scores into the probabilities
- We can use a greedy strategy: peek the word with the highest probability



# Recap of training

## The loss function

- Cross-entropy compares gold standard one-hot encodings to softmax outputs: $CE(p, q) = \sum_{x \in V} p(x) \log q(x)$

| | a | | | | machine | | | zoo |
|---|---|---|---|---|---|---|---|---|
| gold standart | 0 | 0 | 0 | ... | 1 | ... | 0 | 0 |
| softmax output | 0.1 | 0 | 0.1 | ... | 0.6 | ... | 0.2 | 0 |

$V_{target}$

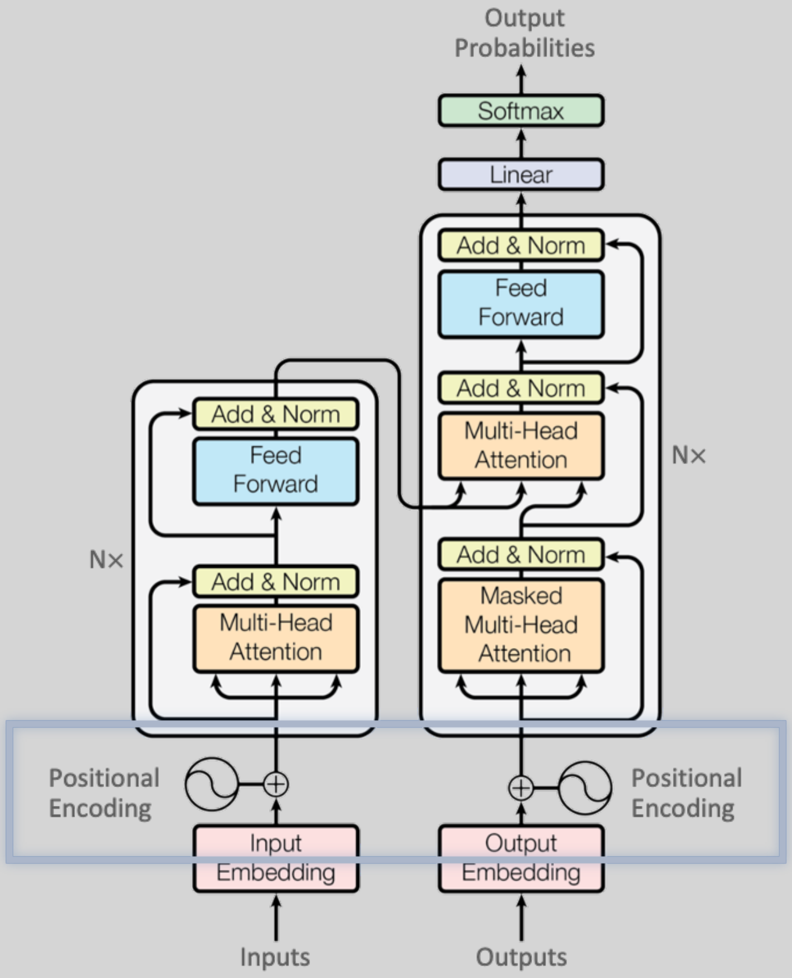
# Positional encodings

- Word embeddings are summed up with positional encodings:

$$x_i = x_i + pe_i$$

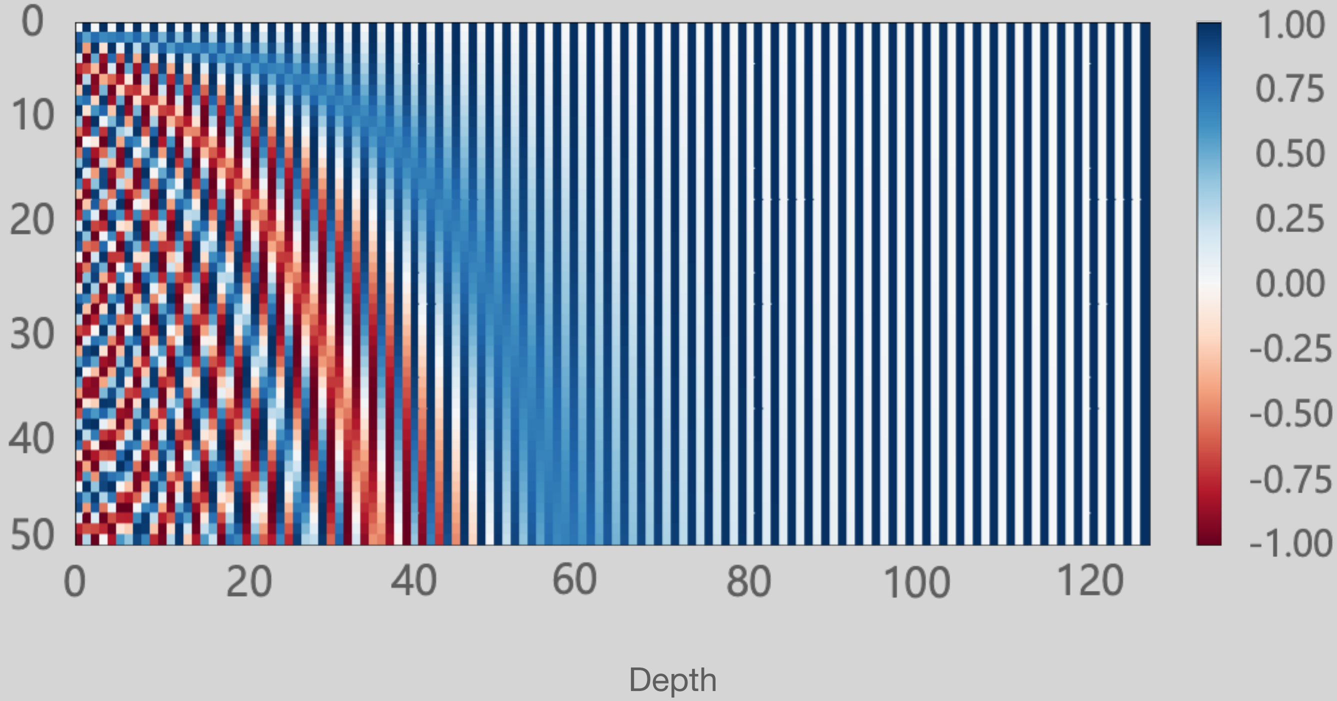
- Non-trainable sin/cos positional encodings

$$pe^{(i)} = \begin{cases} \sin(\omega_k, t) \text{ if } i = 2k \\ \cos(\omega_k, t) \text{ if } i = 2k+1 \end{cases}, \text{ where } \omega_k = \frac{1}{10000^{2k/d}}$$



# Positional encodings

## 128 dim positional encodings for a sequence of length 50

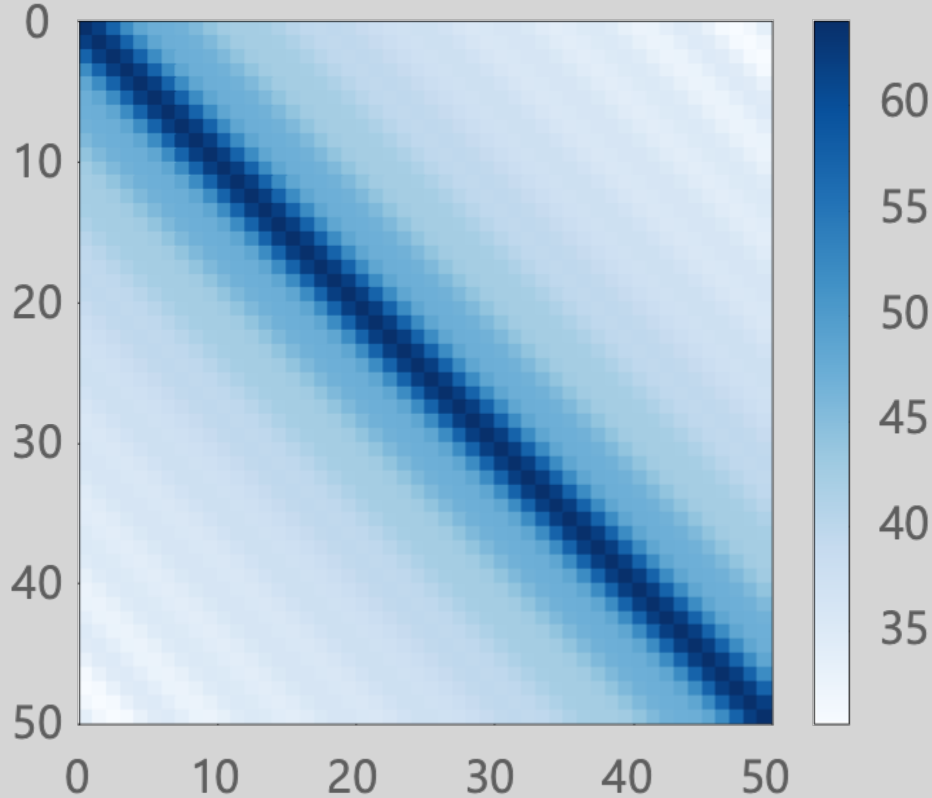





*https://kazemnejad.com/blog/transformer_architecture_positional_encoding/*

# Positional encodings

## 128 dim positional encodings for a sequence of length 50



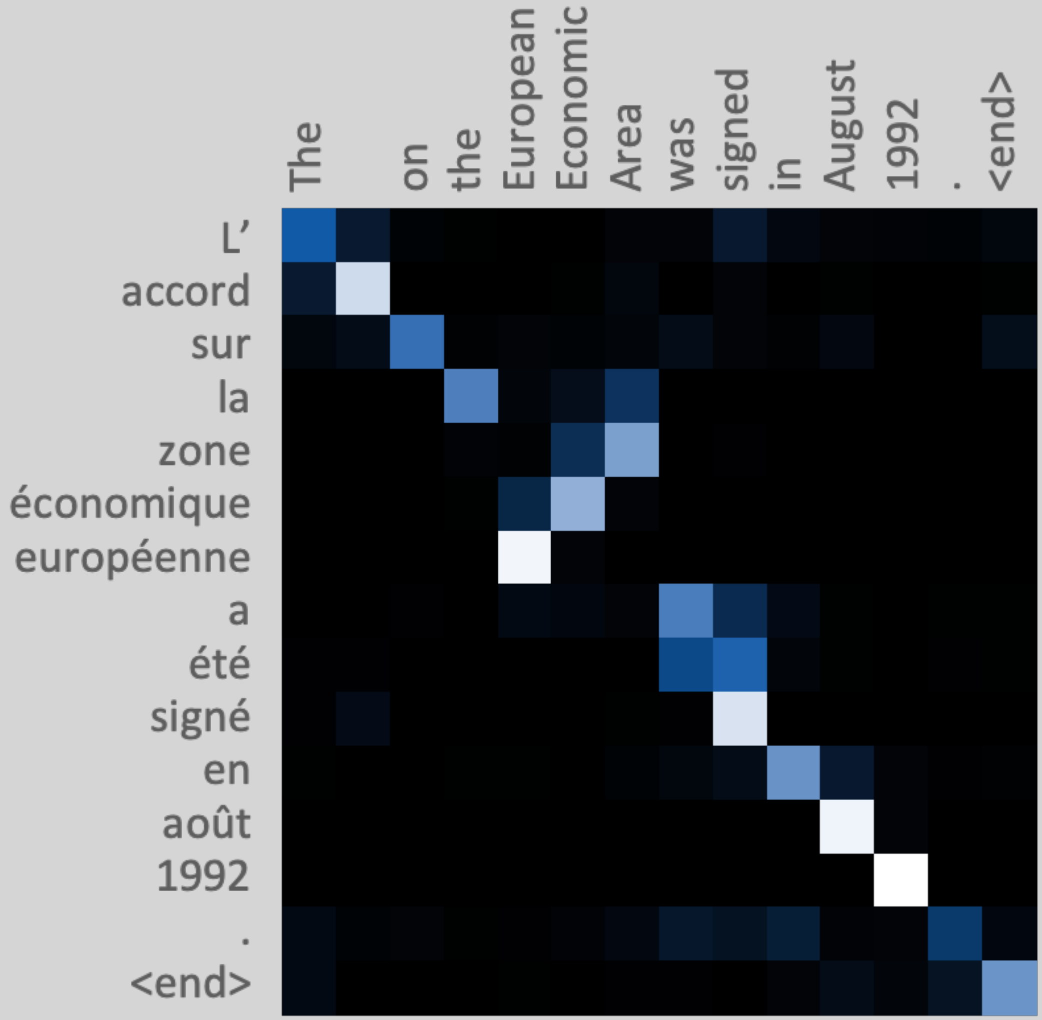



*https://kazemnejad.com/blog/transformer_architecture_positional_encoding/*

# Random facts about MT

# Attention maps

## The Transformer learns word alignment



# Quality metrics in machine translation

## BLEU (bilingual evaluation understudy)

- BLEU uses a modified form of precision to compare a candidate translation against multiple reference translations.
- BLEU computes the same modified precision metric using n-grams.
- BLEU's output is always a number between 0 and 1. The higher the value is, the better.
- BLEU correlates well with human judgements.
- BLEU is non differentiable and can not be optimised directly when the model is trained.



*https://en.wikipedia.org/wiki/BLEU*

# Gender bias in NMT systems

| Finish | English |
| --- | --- |
| Hän on lääkäri | He is a doctor |
| Hän on sairaanhoitaja | She is a nurse |

- State-of-the-art NMT systems suffer from gender biases

# Machine translation

## Research directions

- Translation from many languages to many languages using one model only
- Non-autoregressive models which can generate an output sequence simultaneously
- Reinforcement learning to optimise BLEU
- Training without parallel data
- Detection and removal of ethical biases

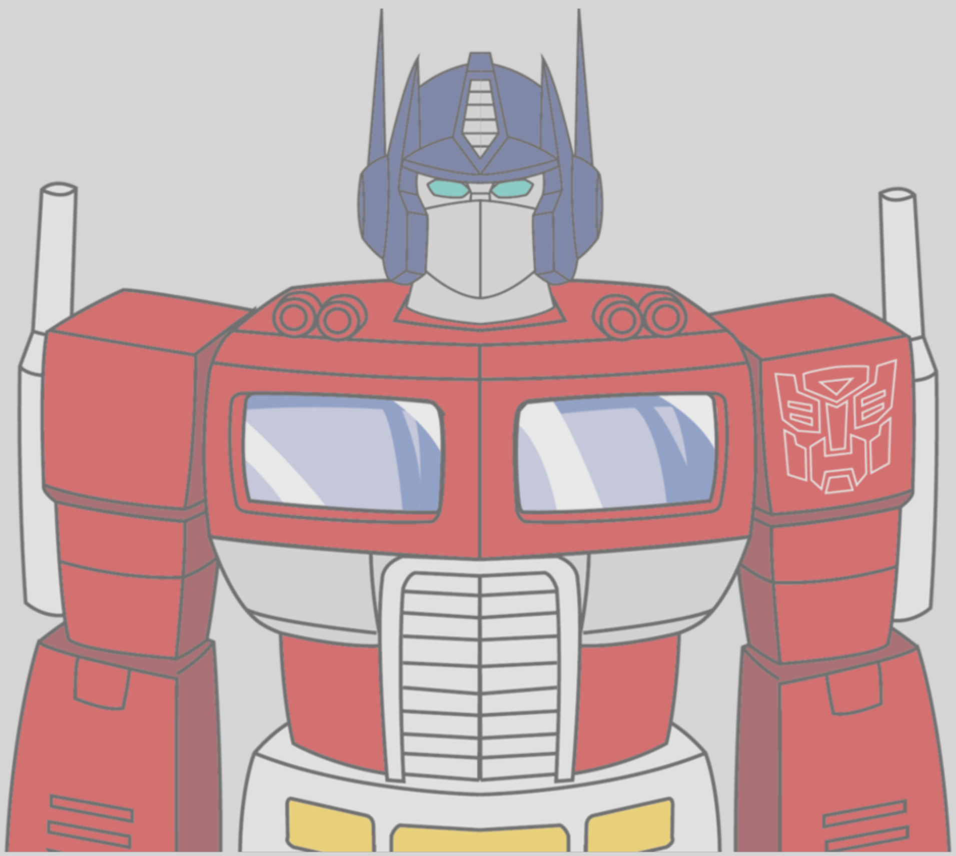
# Other related research directions

- Other applications in NLP, including speech-to-text and text-to-speech transformations
- Transformers in Vision and for Time Series
- Interpretation of learnt attention maps
- Are convolutional nets and Transformers related?
- Making Transformers smaller: distillation, pruning, quantization

# Takeaways

- Many tasks can be formulated as sequence-to-sequence problems
- Transformers are great for such tasks and are widely used now not only for machine translation
- We love Transformers because they gain better results and are faster than many other architectures

# Thank you for your *attention!*



# References


- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In *Advances in neural information processing systems*, pp. 5998-6008. 2017.

- Tomalin, Marcus, Bill Byrne, Shauna Concannon, Danielle Saunders, and Stefanie Ullmann. "The practical ethics of bias reduction in machine translation: Why domain adaptation is better than data debiasing." *Ethics and Information Technology* (2021): 1-15.