

Transformer

Introduced at "Attention is all you need" by Ashish Vaswani et al. (2017)

Outline

Story of attention

- sequence to sequence models [Sequence to Sequence Learning with Neural Networks, by Sutskever et. al. , 2014]
- attention mechanism [Neural Machine Translation by Jointly Learning to Align and Translate, Bahdanau et. al., 2015]
- self-attention [Long Short-Term Memory-Networks for Machine Reading, Cheng et. al., 2015]


Transformer

- model architecture [Attention Is All You Need, Vaswani et. Al., 2017]
- multi-head attention layer

What is attention?

Example 1

The girl who is playing with the ball is wearing a jacket.

The **girl** who is **playing** with the  is wearing a jacket.

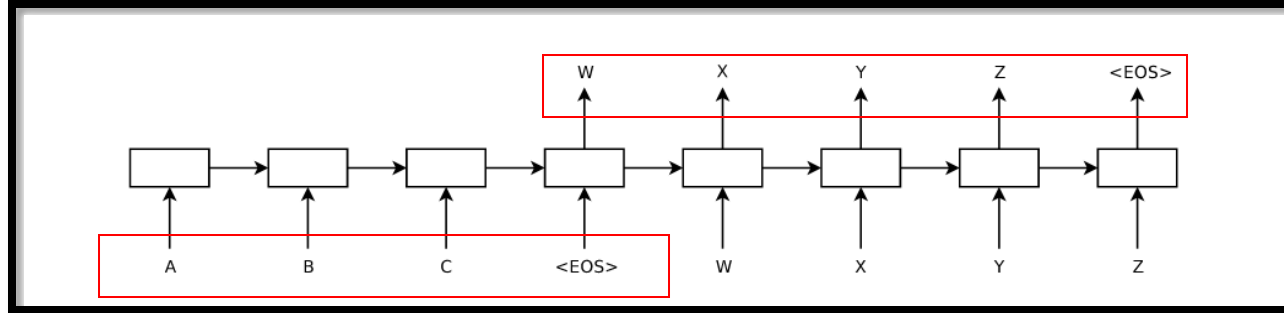
The **girl** who is playing with the ball is **wearing** a .

Example 2



Story of attention mechanism:

seq2seq modeling



Task:

Source sequence → target sequence

- Introduced for language modeling

Sequence to Sequence Learning with Neural Networks, by Sutskever et. al., 2014

- Found application at:

- * Machine translation (audio/text)
- * QA dialogue generation
- * Image caption generation

Method:

- Sequential modeling (RNN/CNN)
- Encoder-decoder architecture with fixed-length context vector

Limitations:

- No explicit mechanism for reasoning over structure (imposes an inductive bias to the structure of data).
- Not suitable for long sequences.

Story of attention mechanism:

seq2seq modeling

Encoder :

maps input sequence
to a context vector

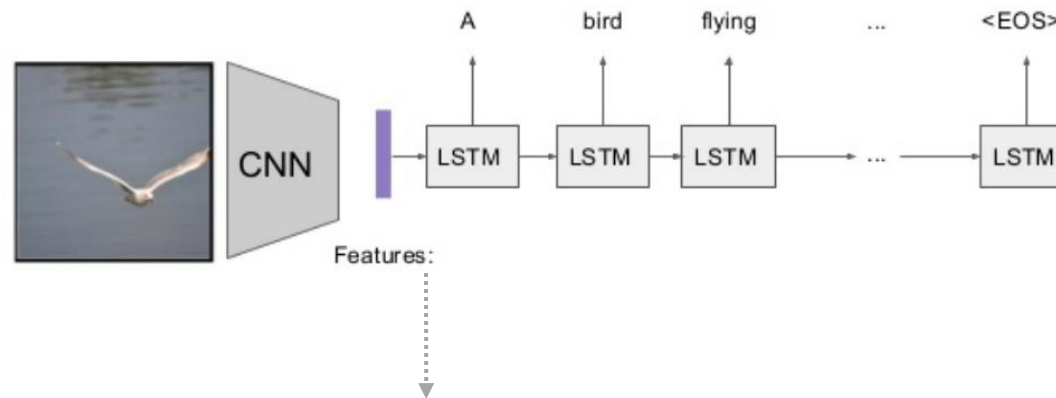
Encoder input: variable-length source sentence



Decoder output: variable-length target sentence

Decoder :

maps the context vector
to another sequence



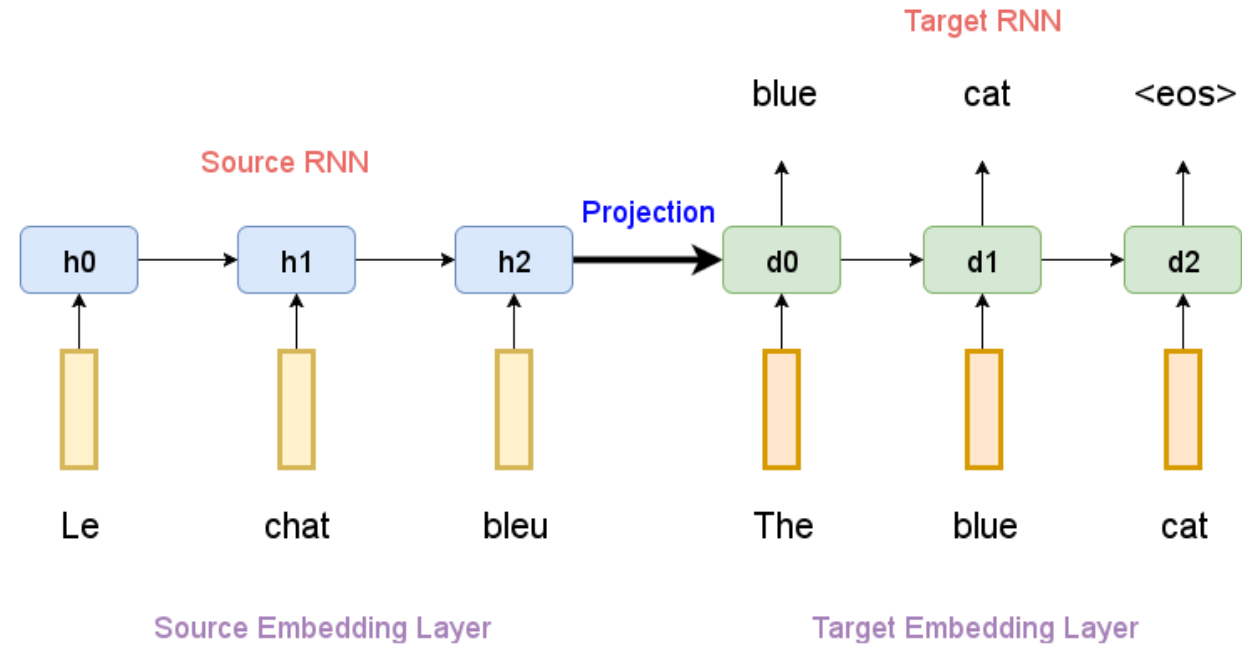
The network compresses all source information into a static fixed-length context vector
And thus all output predictions are based on static output of encoder.

Story of attention mechanism:

seq2seq modeling

initial hidden state is set to the representation v of x_1, \dots, x_T

$$h_t = \text{sigm}(W^{\text{hx}}x_t + W^{\text{hh}}h_{t-1})$$
$$y_t = W^{\text{yh}}h_t$$



The model tries to estimate conditional probability of:

$$p(y_1, \dots, y_{T'} | x_1, \dots, x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, \dots, y_{t-1})$$

Story of attention mechanism:

- Introduced for NMT :

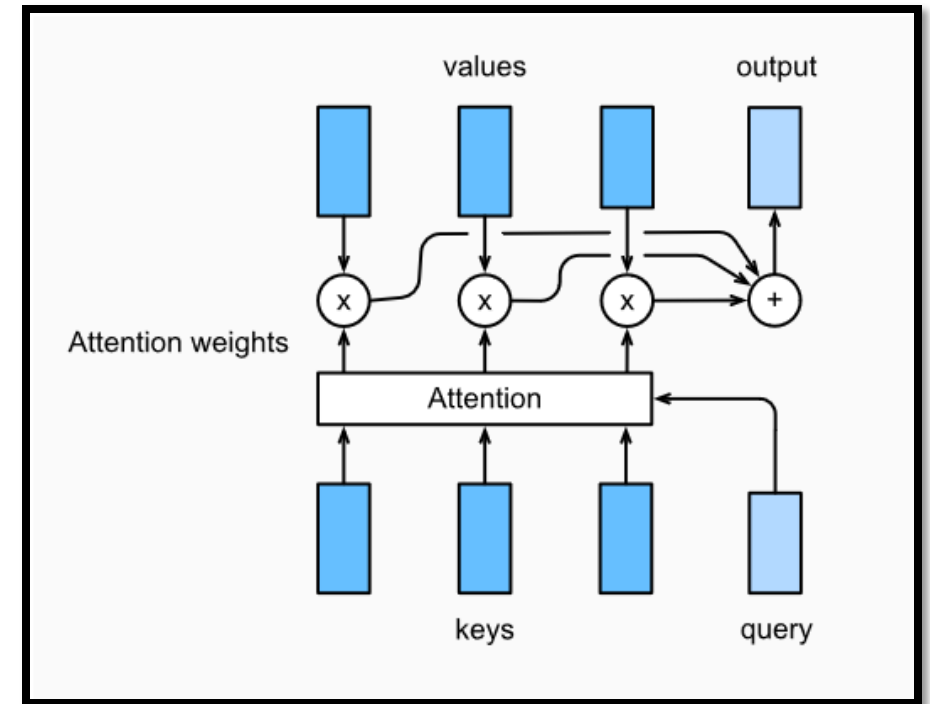
To automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word. The model then predicts a target word based on the context vectors associated with these source positions and all the previous generated target words.

Neural Machine Translation by Jointly Learning to Align and Translate,
Bahdanau et. al., 2015

Advantages:

- The model automatically finds correspondence between source and target sequences (alignment)
- Suitable for long sequences

attention layer



The attention layer returns an output based on input query and its memory.

Story of attention mechanism:

$$p(y_i | y_1, \dots, y_{i-1}, \mathbf{X}) = g(y_{i-1}, s_i, c_i),$$

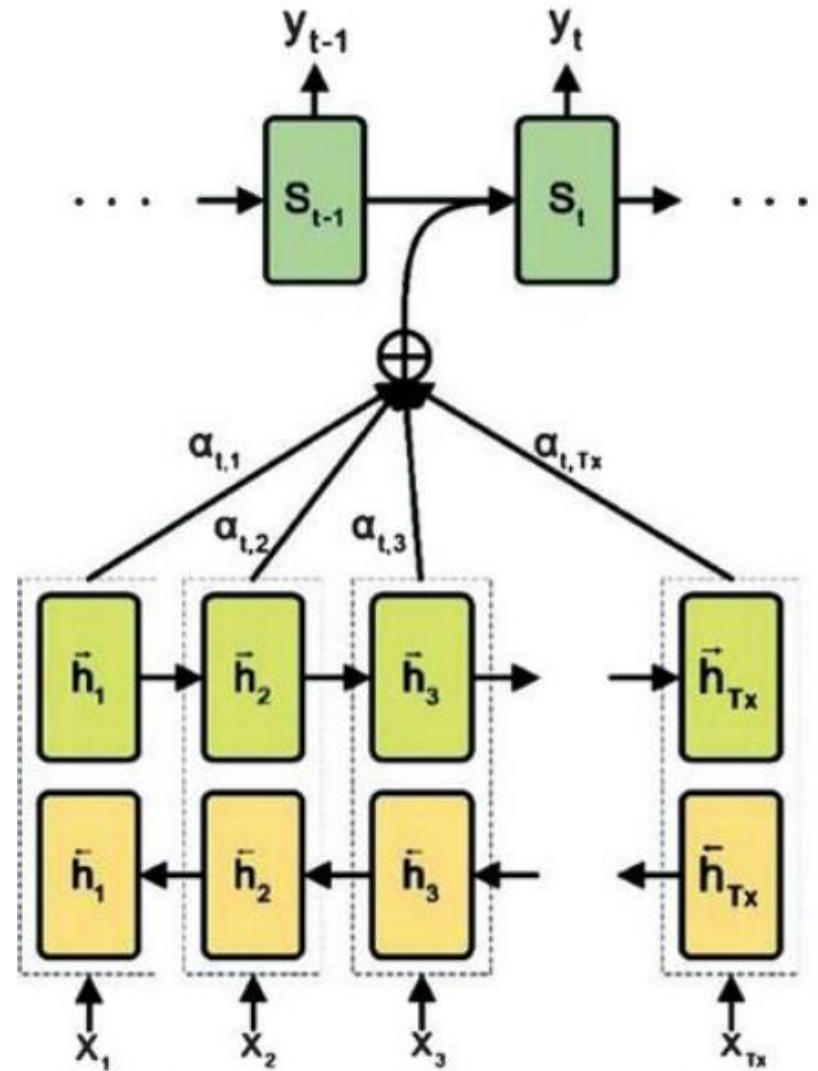
$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})},$$

$$e_{ij} = a(s_{i-1}, h_j)$$

attention layer



Story of attention mechanism:

- Introduced for "Machine Reader" (understanding the text):

Offers a way to weakly induce relations among tokens by modifying the standard LSTM structure by replacing the memory cell with a memory network (LSTMN).

Long Short-Term Memory-Networks for Machine Reading, Cheng et. al., 2015

- Found application at:

- * language modeling
- * sentiment analysis
- * ?

Self-attention

Red
represents
current token

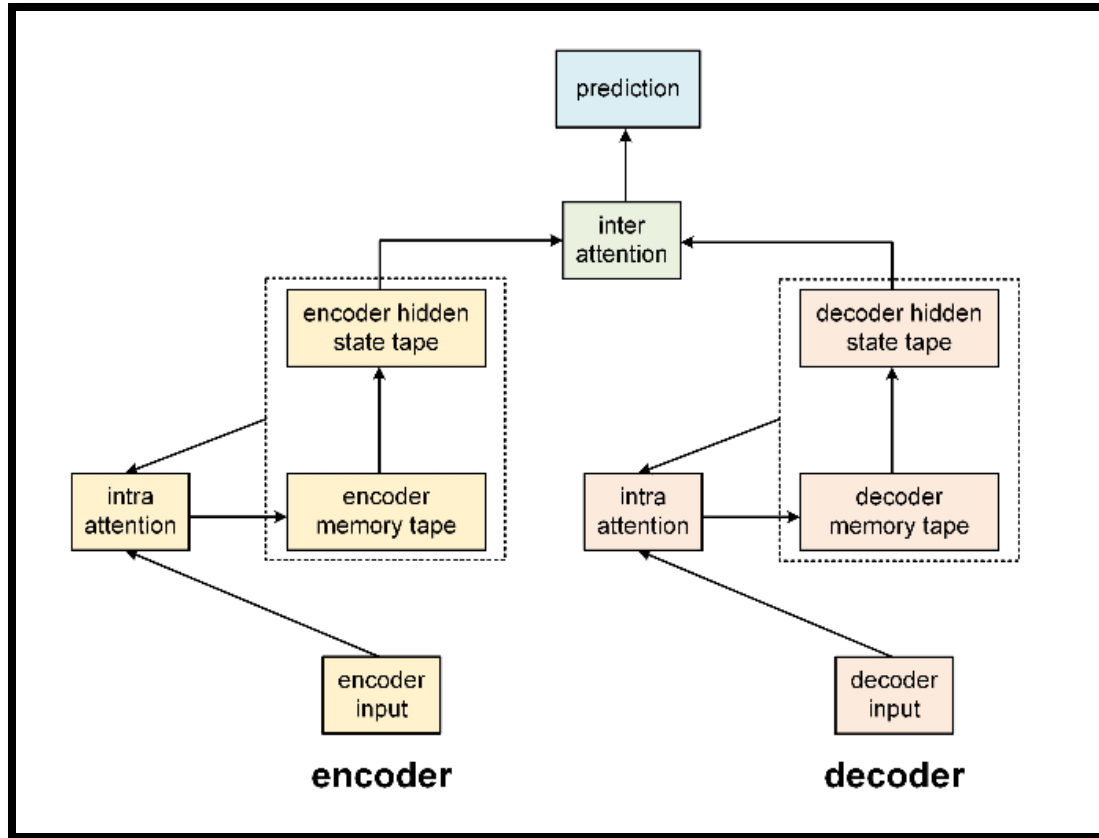
Blue
represents
memories

The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .

- Text is processed incrementally while learning which past tokens and to what extent they relate to the current token.
- Feature: the model induces undirected relations among tokens
- Method: attention is added within a sequence encoder

Story of attention mechanism:

attention layer



Self-attention, (also called intra-attention), is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence.

Shallow attention fusion, Cheng et. al, 2015

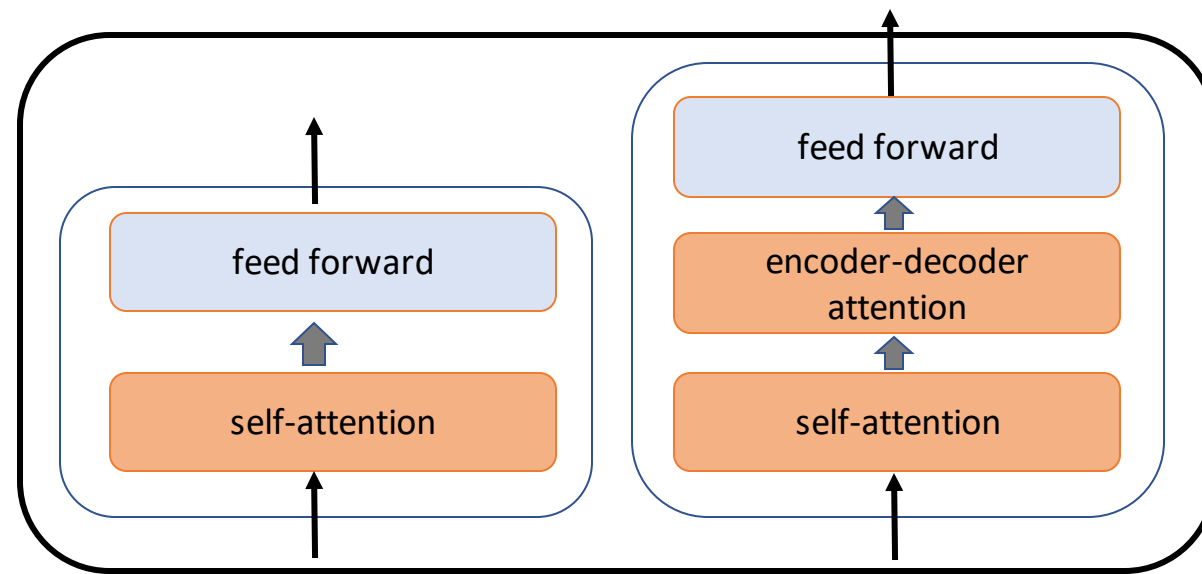
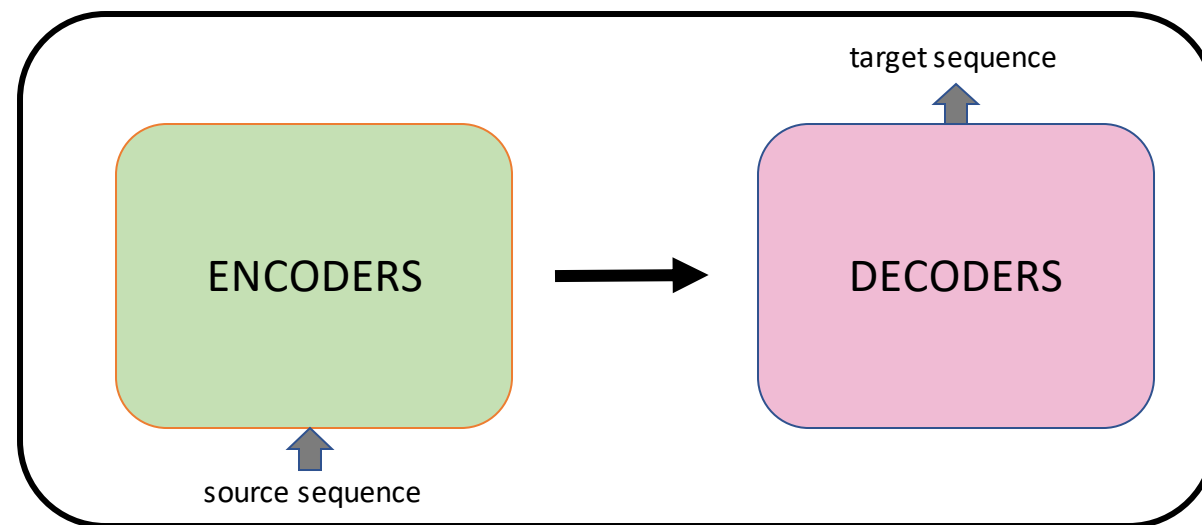
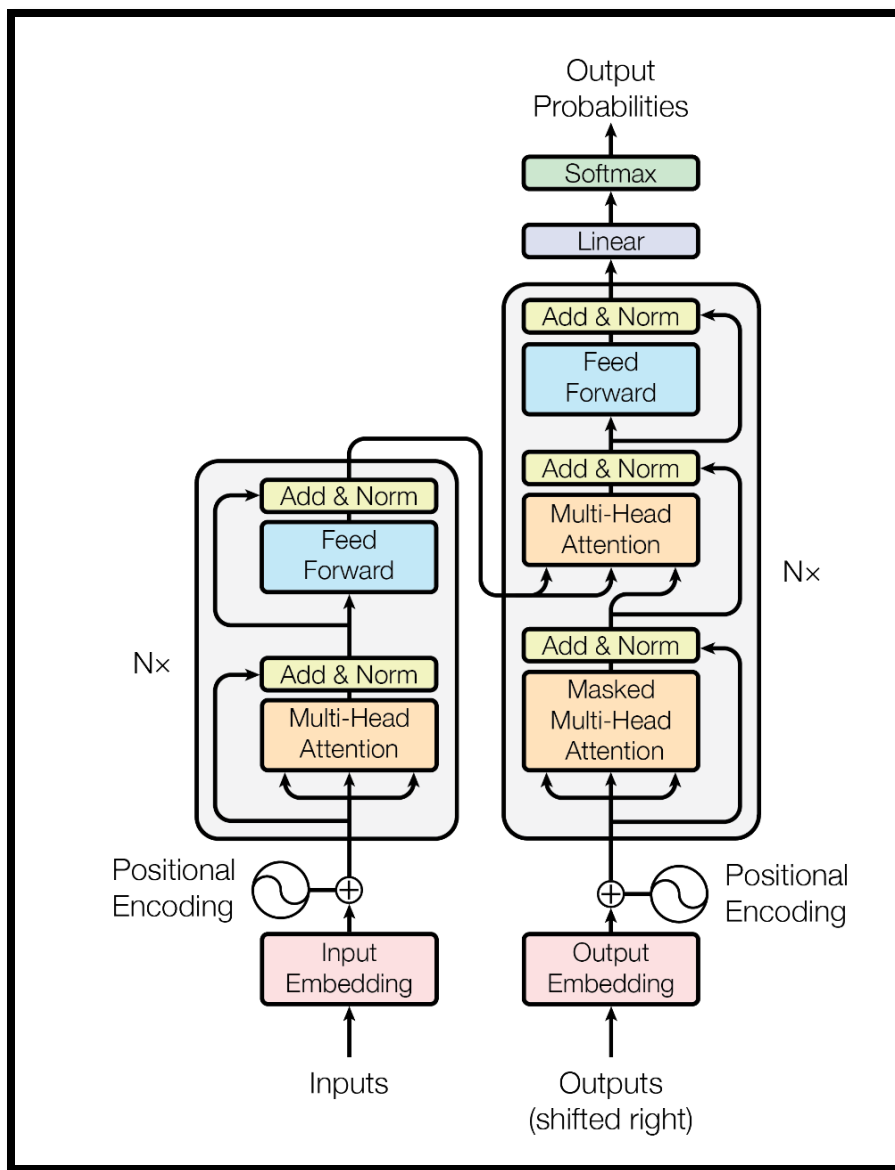
Story of attention mechanism:

Aligning functions

Name	Alignment score function	Citation
Content-base attention	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \text{cosine}[\mathbf{s}_t, \mathbf{h}_i]$	Graves2014
→ Additive(*)	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\mathbf{s}_t; \mathbf{h}_i])$	Bahdanau2015
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \mathbf{s}_t)$ Note: This simplifies the softmax alignment to only depend on the target position.	Luong2015
General	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{s}_t^\top \mathbf{W}_a \mathbf{h}_i$ where \mathbf{W}_a is a trainable weight matrix in the attention layer.	Luong2015
→ Dot-Product	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \mathbf{s}_t^\top \mathbf{h}_i$	Luong2015
Scaled Dot-Product(^)	$\text{score}(\mathbf{s}_t, \mathbf{h}_i) = \frac{\mathbf{s}_t^\top \mathbf{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.	Vaswani2017

Transformer

(multi-head self-attention)

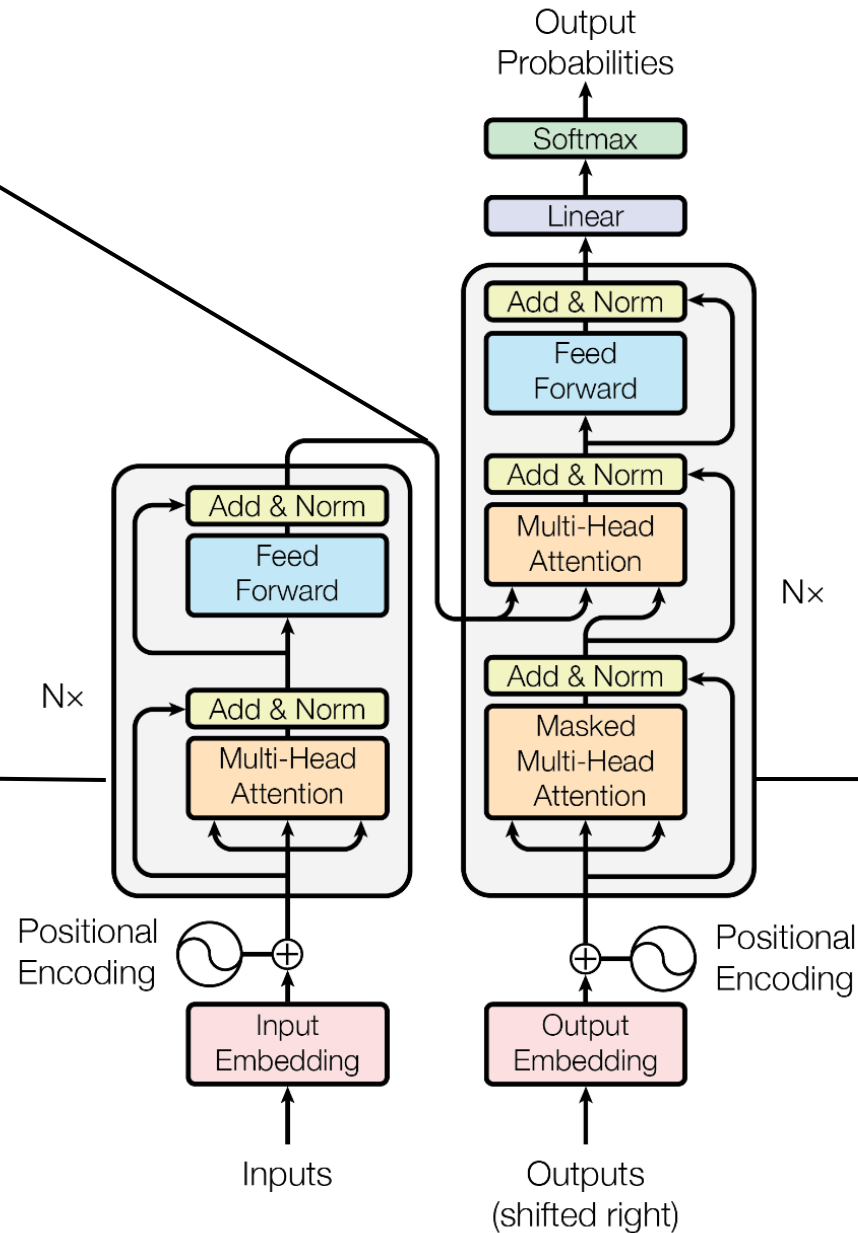


encoder-decoder
attention

Relying entirely on
self-attention to
compute
representations of
its input and output!

Auto-regressive :
at each time step the
model is consuming
the previously
generated symbols as
additional input!

Self-attention:
learning long-
range
dependencies

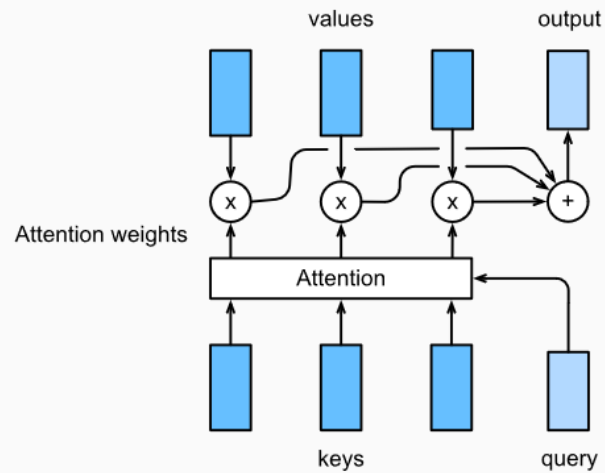


Attention layer

Query vector?

Value vector?

Key vector?



Attention input:

- Q vector (set of queries)
- K vectors (set of keys)
- V vector (set of values)

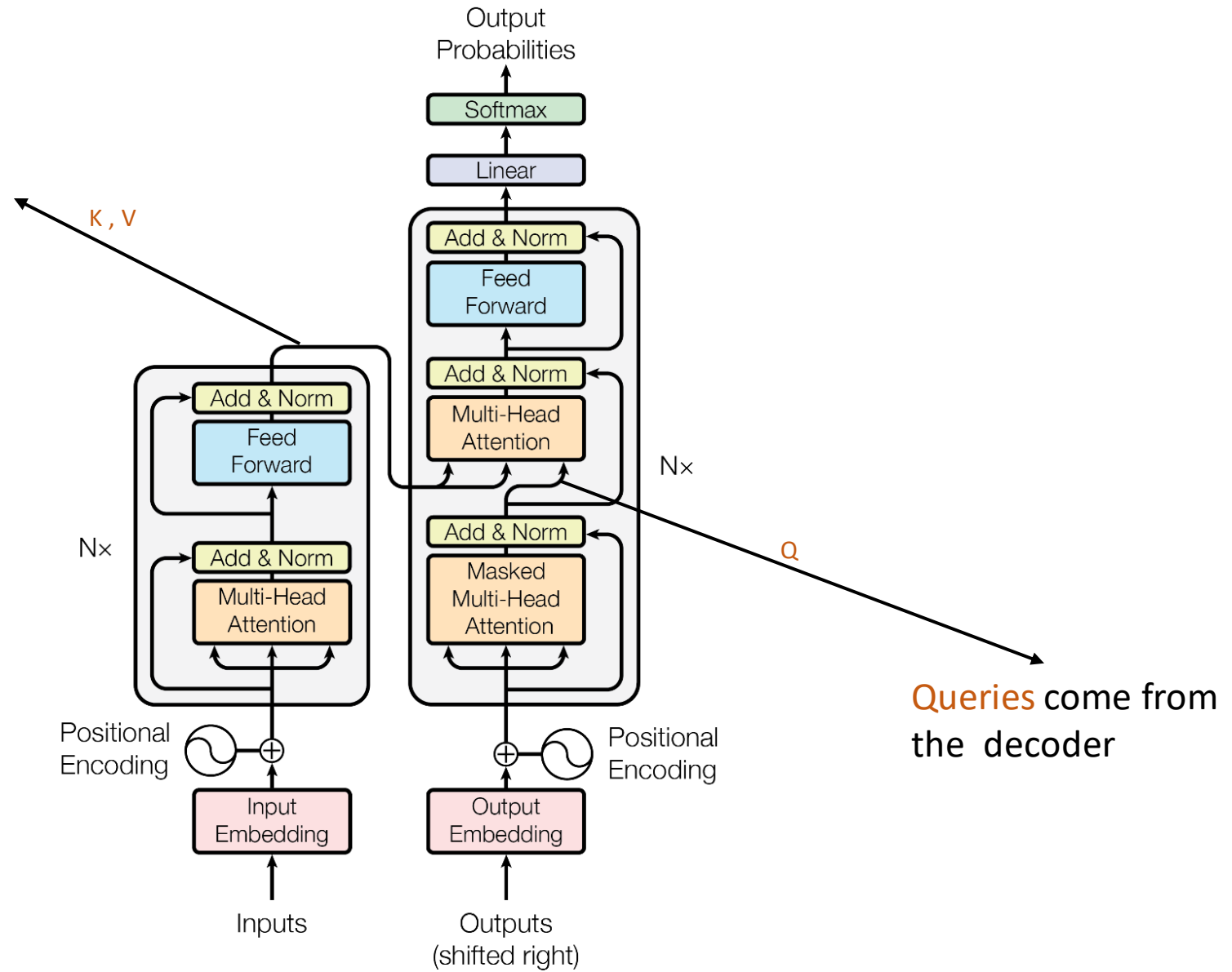


Attention output:

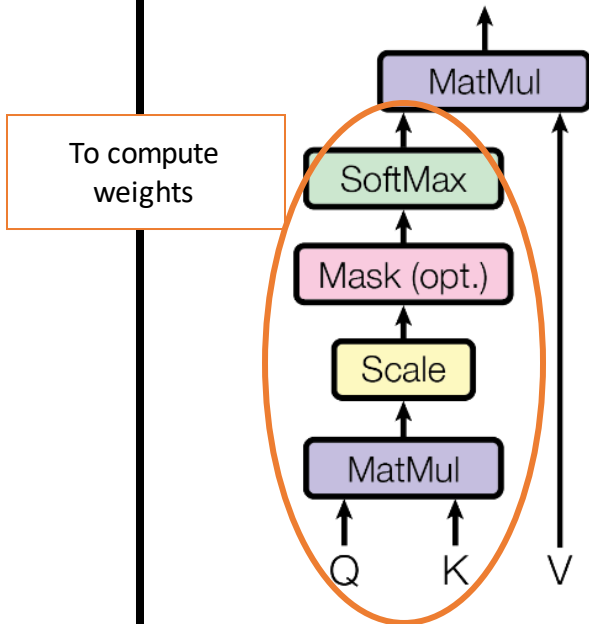
- weighted sum of values

Memory Keys and Values

Come from encoded representation of source sequence



Scaled Dot-Product Attention

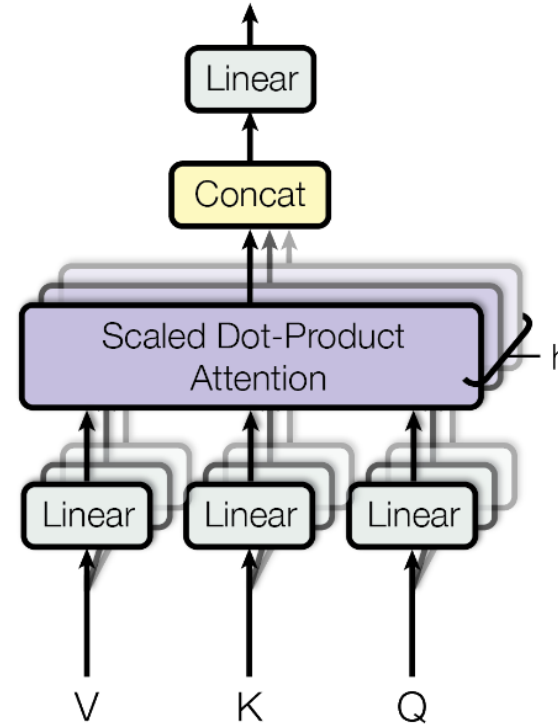


To compute weights

Weighted sum of values

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

Multi-Head Attention



back to $d(\text{model}) = 512$

8 parallel channels of size 64

- attention function is performed in each parallel channel separately
- allows the model to jointly attend to information from different subsets
- works better than averaging
- total computation cost is smaller

$d(\text{model}) = 512$

Summary

- Attention helps to focus on relatively important part of sequential data.
- Self-attention is an attention mechanism that relates different parts of a single sequence in order to make a representation of that sequence.
- Transformer relies entirely on self-attention to compute representation of input and output of the network without using recurrent or convolution layers.

Thank you!
