

Transformers

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

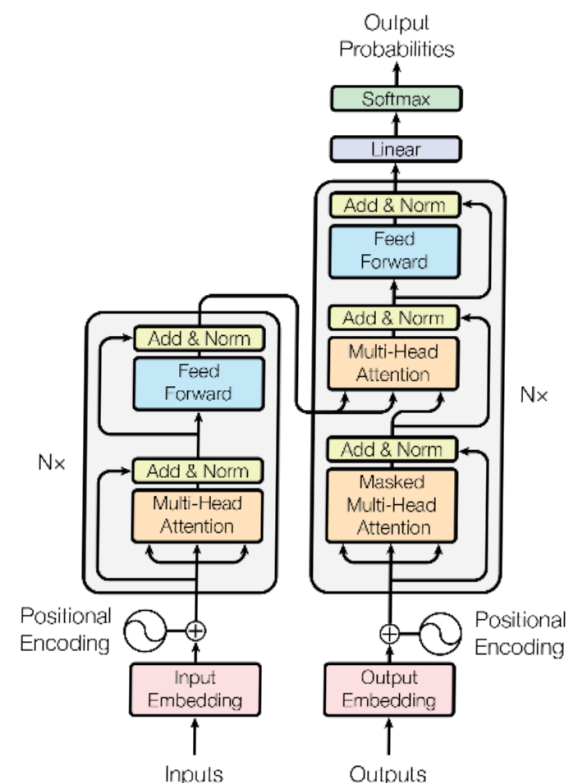
Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Lukasz Kaiser*
Google Brain
lukaszkaizer@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.



Shouvik Sarkar
192CS022

Bag Of Words

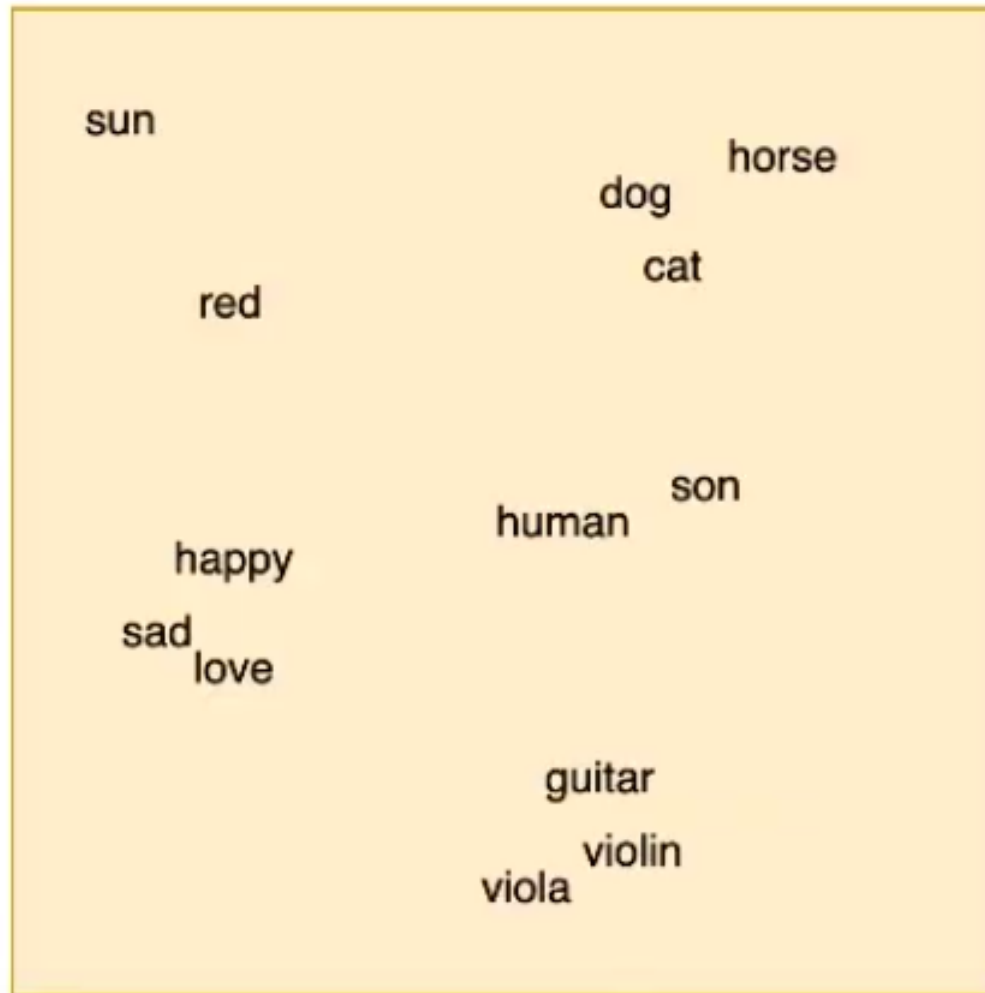
I had a **good** day

I had a **great** day

I had a **had** day

	I	had	a	good	great	day
I	1	0	0	0	0	0
had	0	1	0	0	0	0
a	0	0	1	0	0	0
good	0	0	0	1	0	0
great	0	0	0	0	1	0
day	0	0	0	0	0	1

Embedding space



Word Embeddings

happy	0.5	0.45	0.76
sad	0.45	0.45	0.82
red	0.01	0.9	0.2

Pretrained embeddings can be downloaded using libraries like FastText

Positional Encoding

Impart positional information to the vectors

Sara's **dog** is cute

Sara is cute like a **dog**

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

So far

For each word

Embedding (vector)

+

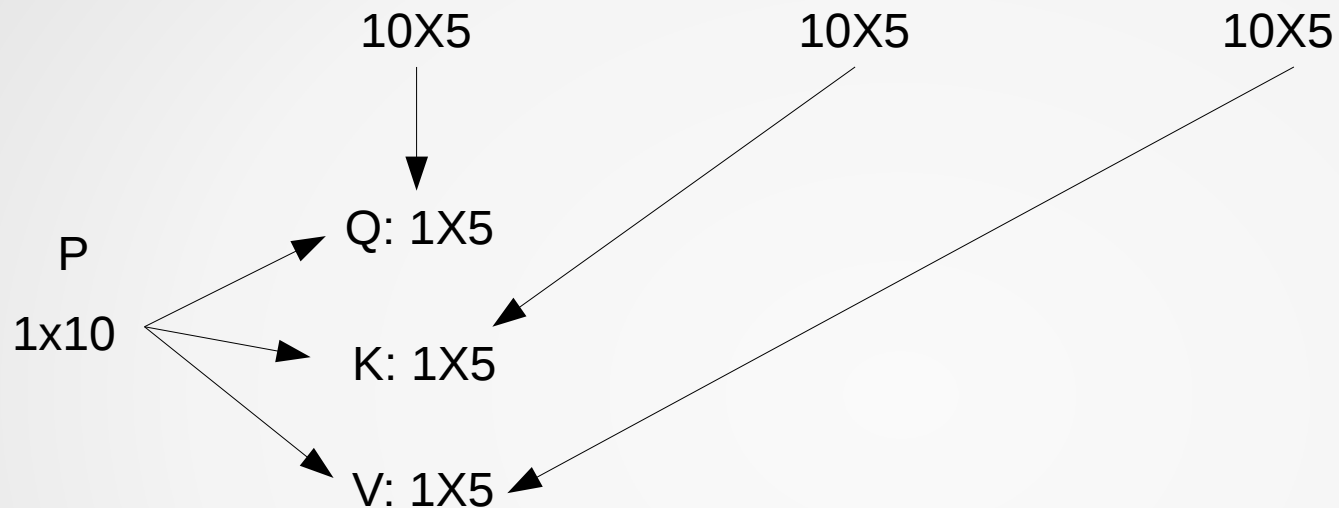
Positional Encoding (operation)

=

Context with position info

To be done for both encoder and decoder

Queries, Keys and Values

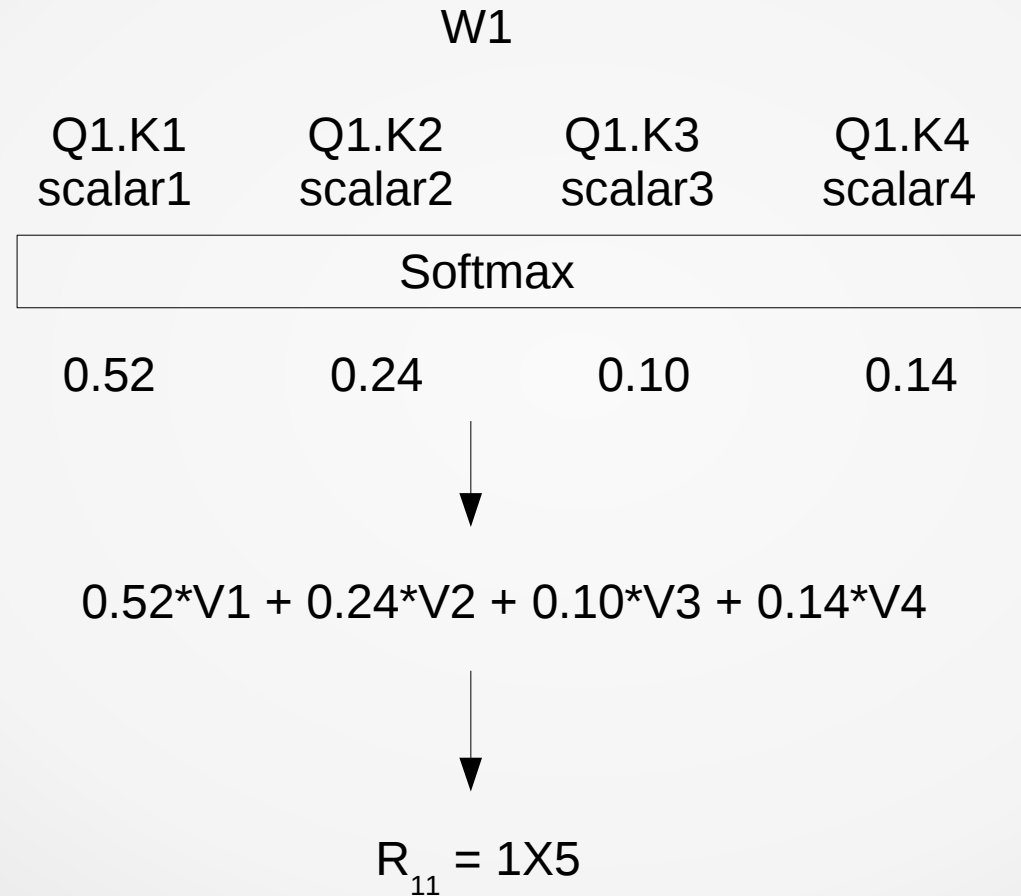


Do the same for all word vectors
(Let's assume a 4 word sentence)

Queries, Keys and Values

	Word1	Word2	Word3	Word4
Positionally Encoded Vectors	P1	P2	P3	P4
Queries	Q1	Q2	Q3	Q4
Keys	K1	K2	K3	K4
Values	V2	V2	V3	V4

Self - Attention



Self - Attention

- Computes attention vector for each word

The day was good

0.7	0.6	0.54	0.32	0.43
-----	-----	------	------	------

The day was good

0.8	0.9	0.6	0.5	0.4
-----	-----	-----	-----	-----

The day was good

.43	0.52	0.76	0.54	0.32
-----	------	------	------	------

The day was good

0.45	0.23	0.54	0.67	0.49
------	------	------	------	------

Multi Head Attention

W1 \longrightarrow $\begin{matrix} R_{11} \\ R_{12} \end{matrix}$ \longrightarrow Concat (R_{11}, R_{12}) \longrightarrow 1X10 (R1)

W2 \longrightarrow $\begin{matrix} R_{21} \\ R_{22} \end{matrix}$ \longrightarrow Concat (R_{21}, R_{22}) \longrightarrow 1X10 (R2)

W3 \longrightarrow $\begin{matrix} R_{31} \\ R_{32} \end{matrix}$ \longrightarrow Concat (R_{31}, R_{32}) \longrightarrow 1X10 (R3)

W4 \longrightarrow $\begin{matrix} R_{41} \\ R_{42} \end{matrix}$ \longrightarrow Concat (R_{41}, R_{42}) \longrightarrow 1X10 (R4)

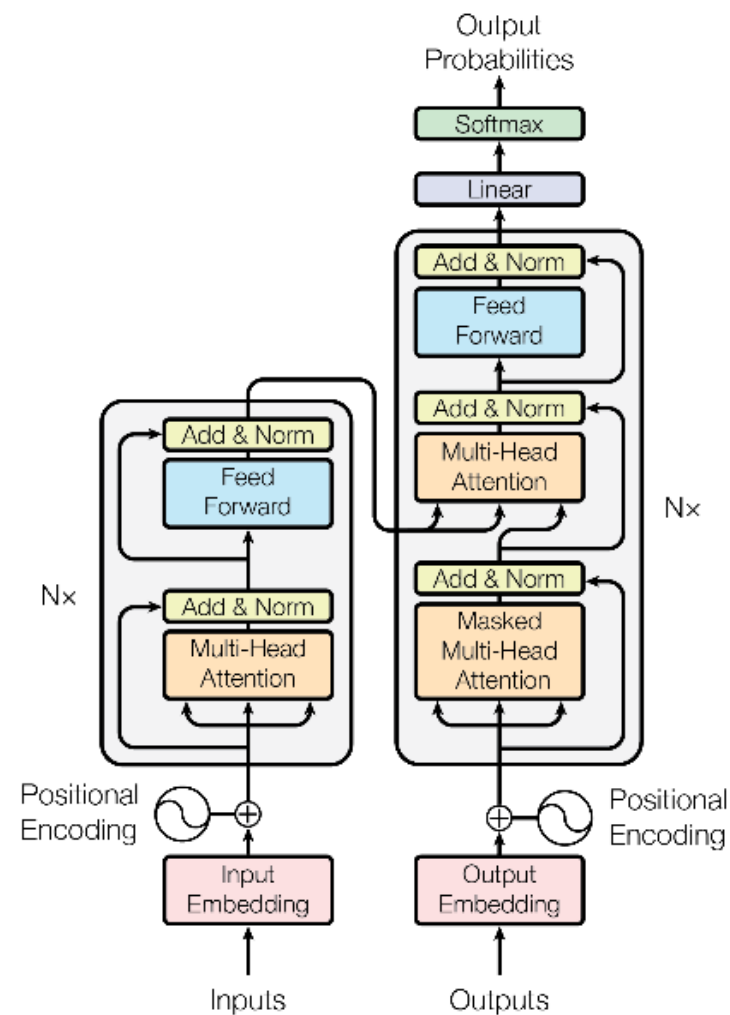
Projection

$$\begin{bmatrix} R \\ 1 \times 10 \end{bmatrix} \begin{bmatrix} W_0 \\ 10 \times 10 \end{bmatrix} = \begin{bmatrix} Pr \\ 1 \times 10 \end{bmatrix}$$

Layer Normalisation

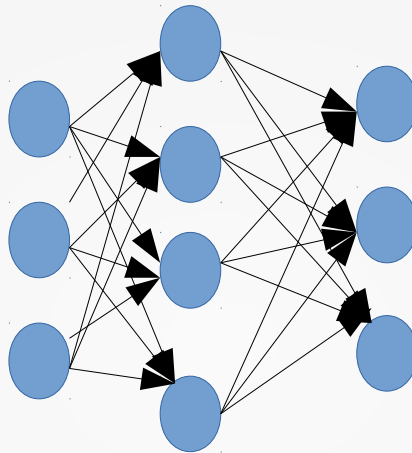
P is the positionally encoded vector fed into the self-attention layer

Layer Normalisation ($P + P_r$)



Feed Forward, Add and Normalize

P_{r_i}
Input size = size of a P_r vector



Output is same size as Input



Layer Normalization ($P_{r_i} + \text{Output}_i$)

Repeat

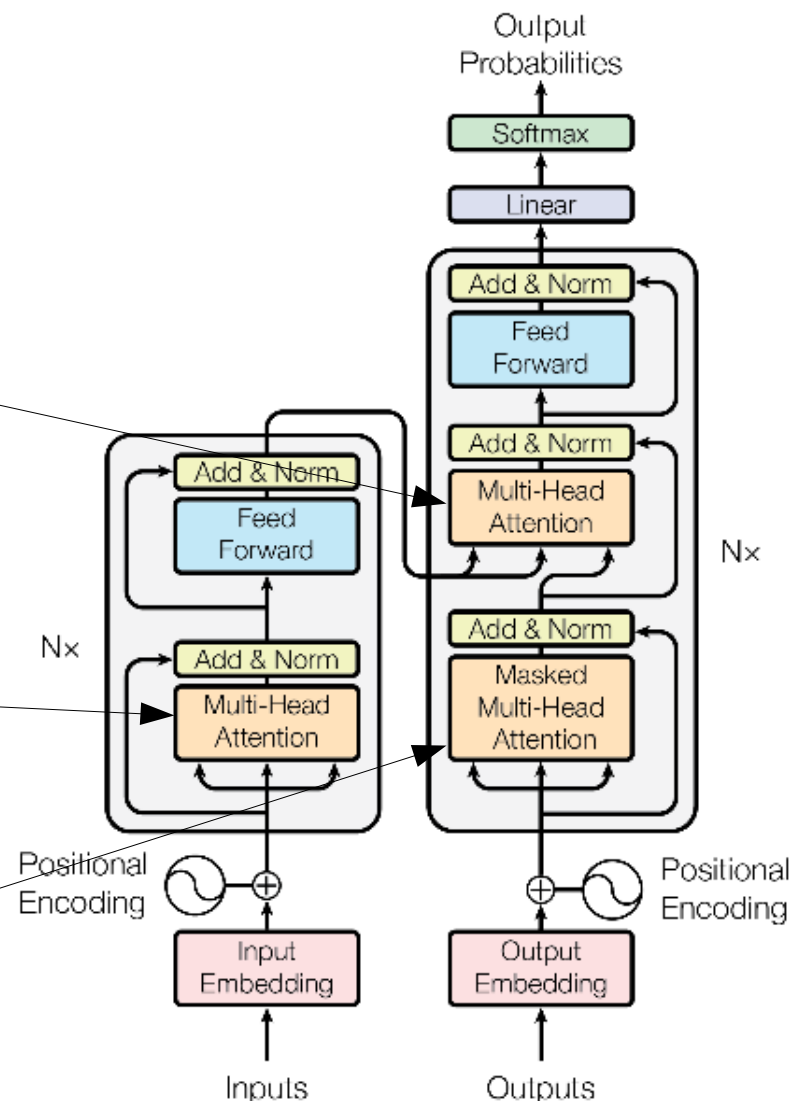
- The operations so far constitute one encoder layer.
- This output becomes the input for the next encoder layer
- Repeat process ($N = 6$)
- The output of the final encoder layer is used in the multi - Head attention layer of the decoder

What is being attended ?

Target Language and
Source Language

Source Language and
Source Language

Target Language and
Target Language



Masked Self Attention in Decoder

- The generation of a word in the target language can pay attention only to the words already generated.
- Since operations are vectorized there has to be a mechanism to mask the positions for the words that will be generated afterwards.

Linear and Softmax Layer in Decoder

Will project onto the vocabulary of the target language thereby giving a probability distribution over the candidate next words.

Two ways of choosing words

- Greedy Approach
- Beam Search

Training

Feed Forward

I	there	go	am	want	come	fast	can	to	will
0.10	0.13	0.12	0.03	0.31	0.01	0.02	0.15	0.04	0.09

Target

0.	0.	0.	0.	0.	1.	0.	0.	0.	0.
----	----	----	----	----	----	----	----	----	----

Loss Function = Sum of differences
Goal is to minimize the loss function