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

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

arXiv Link to the paper

*Published in **Neural Information Processing Systems(NIPS)**, 2017*

*Presented by **Partha Pratim Saha***

19 Feb 2020

Problems that we want to solve using AI

Various real life applications:

- ★ **Machine translation:** Translate a sentence from English to German or French
  - Google translator is using such a model in their production from 2016
- ★ **Language modeling:** Predict next best word when you chat
- ★ **Image Captioning:** Given an image, machine explains about it automatically
- ★ **Advantages**:
  - If your student or client is from another country, AI is helping you in the mentioned scenarios through this model

Examples

1. Group picture of nine tourists and one local on a grey rock with a lake in the background;
2. Five people are standing and four are squatting on a brown rock in the foreground;

More examples: Our ambiguous language

- **"Bob could not put the trophy inside the suitcase because it was too big."**
  - ➔ What does **it** refers to here:**Trophy OR Suitcase?**

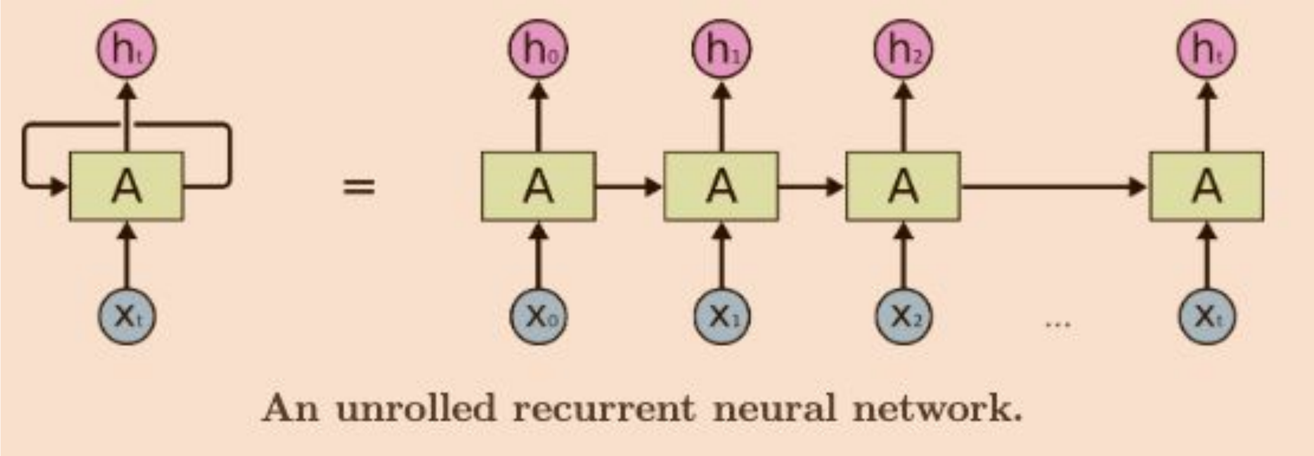
- **“Margarett dropped the plate on the table and broke it."**
  - ➔ What does **it** refers to here:**Table OR Plate?**

- **“The animal didn't cross the street because it was too tired.”**
  - ➔ What does **it** refers to here:**Animal OR Street?**

- **“The animal didn't cross the street because it was flooded.”**
  - ➔ What does **it** refers to here:**Animal OR Street?**

Existing Deep Learning solution (1)

- **RNN (Recurrent Neural Network)** that is used when the output from previous step is fed as input to the next unit as we see in the below picture:

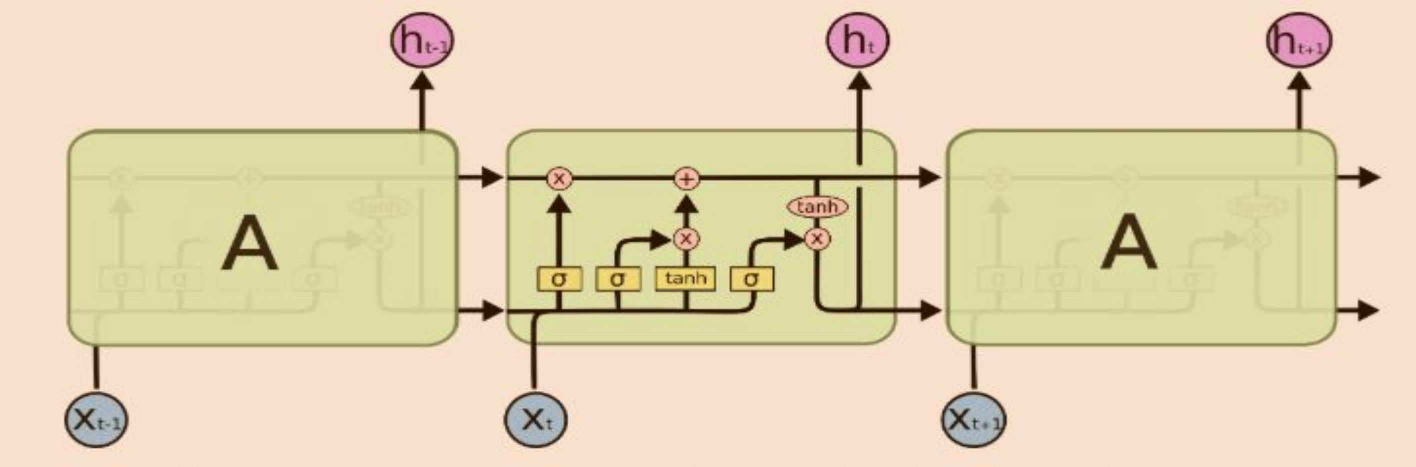


★ **Drawback:**

- Vanishing and exploding gradient.
- Prohibits parallelization within instances
- Next step output depends on previous step

Existing Deep Learning solution (2)

- **LSTM (Long short term memory)** is a variant of RNN that is used to remember some part of the previous dependent words contextually.



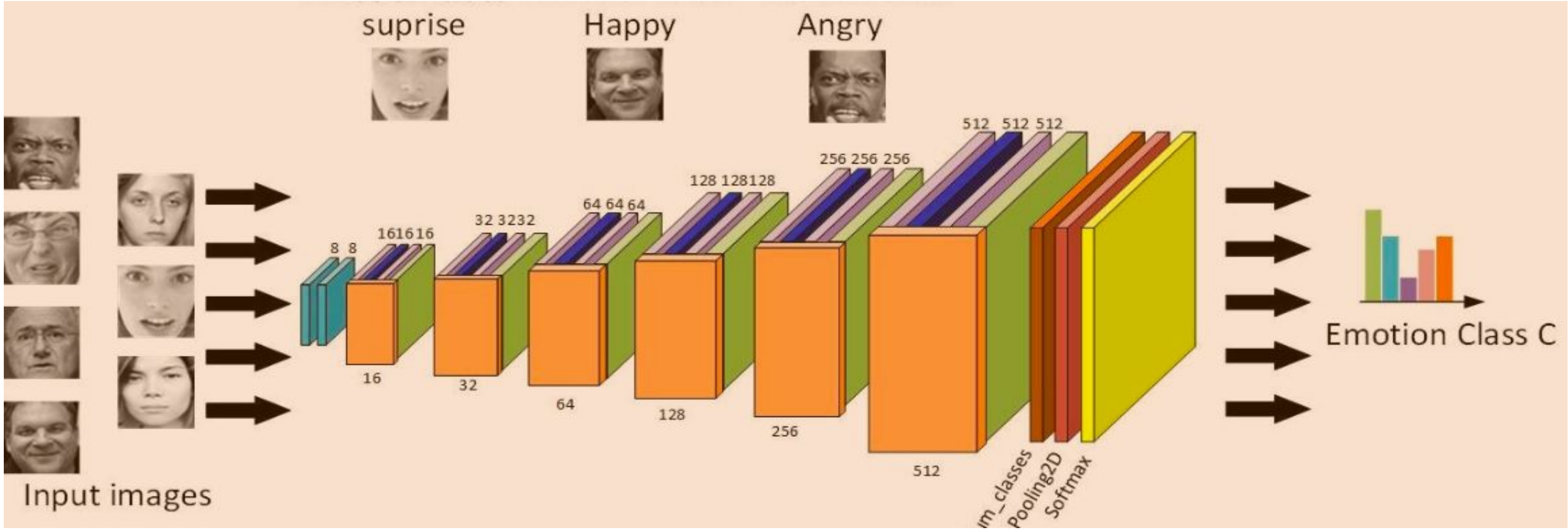
The repeating module in an LSTM contains four interacting layers.

★ **Drawback:**

- Cannot parallelization within instances

Existing Deep Learning solution (3)

- **CNN (Convolution Neural Network)** is used for object identification in a given image (spatial data)



★ **Drawback:**

- Parallelize within layer but NOT across layers in the network
- Slow running time

Proposed solution: Transformer model & some definitions

- **Sequence2Sequence:** A technique that takes a sequence of items like words, letters, features of an images…etc as a vector and outputs another vector as a sequence of items.

- **Encoder**: A function maps an input sequence of symbol representations (X1..Xn) to a sequence of continuous representations Z = (Z1.. Zn)

- **Decoder**: Given a representation Z, the decoder generates an output sequence (Y1..Ym) of symbols one element at a time.

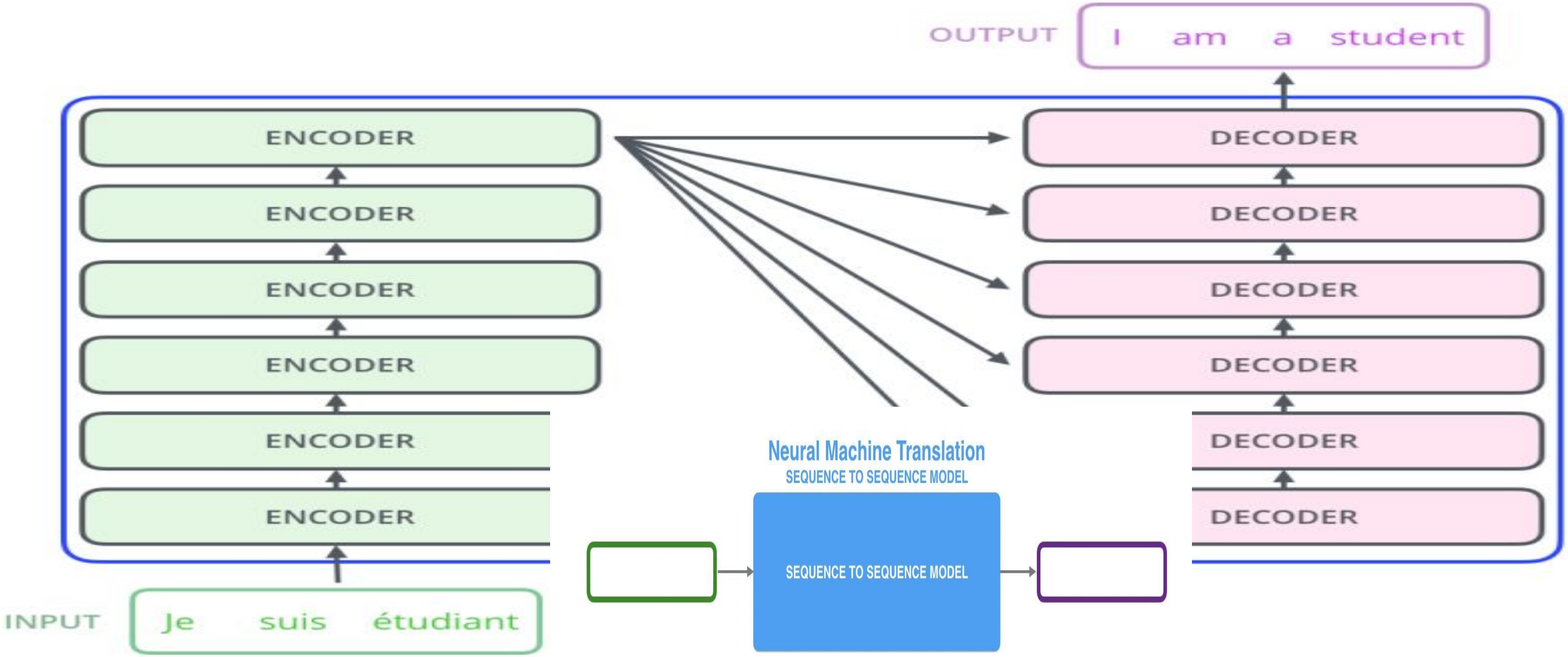
- **Attention**: A function that map a given query, key, value to a probability distribution, where the query, keys, values, and output are vectors.

- **Self Attention:** A method focus on some of the words in the vicinity of the given input sequence

- **Multihead Attention**: Doing self attention mechanism multiple times linearly with different inputs of Q/V/K matrices & have different sets of output matrices to concatenate them together.

- **Output**: Weighted sum of the values, where weight assigned to each value is computed by a function of the query with the corresponding key.

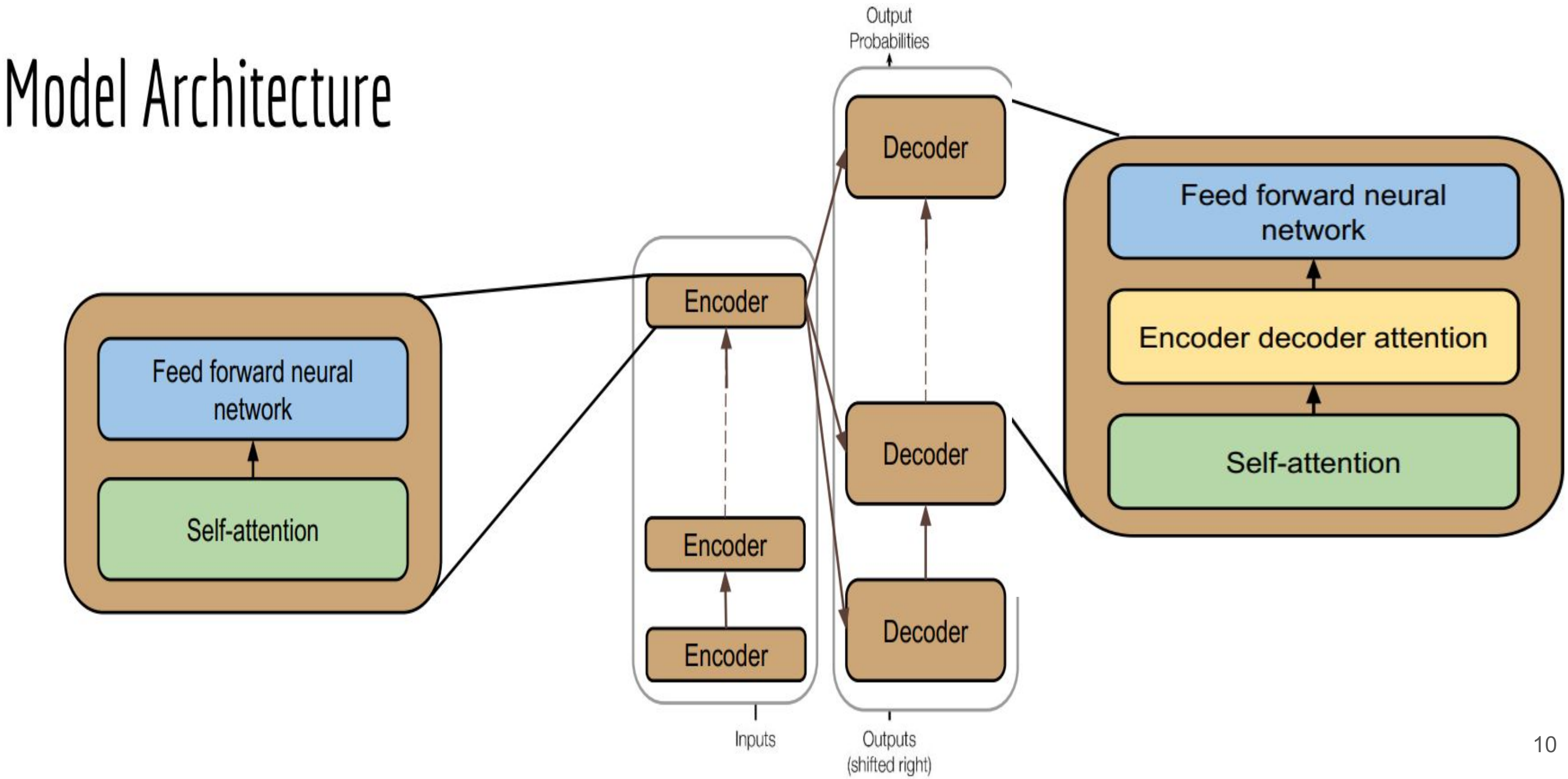
Transformer Model (6 identical encoders and decoders)



★ **Advantages**

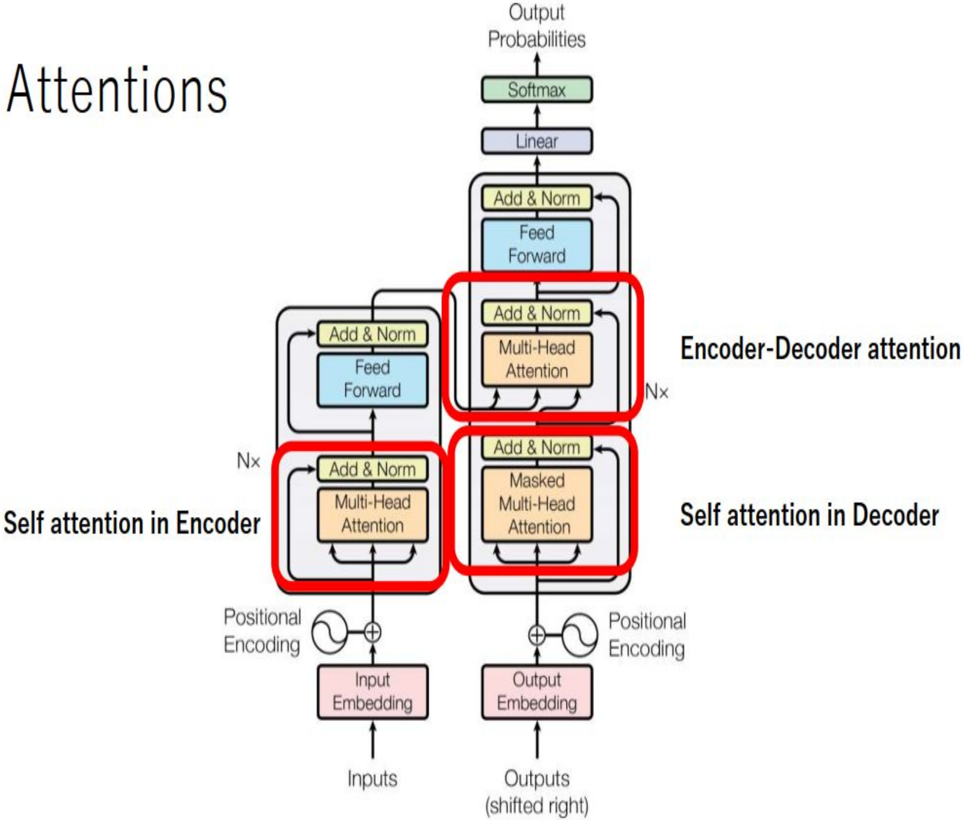
- Capture long range dependencies using self attention
- Sequentiality parallelize within instances

Deep dive into Encoder - Decoder layers

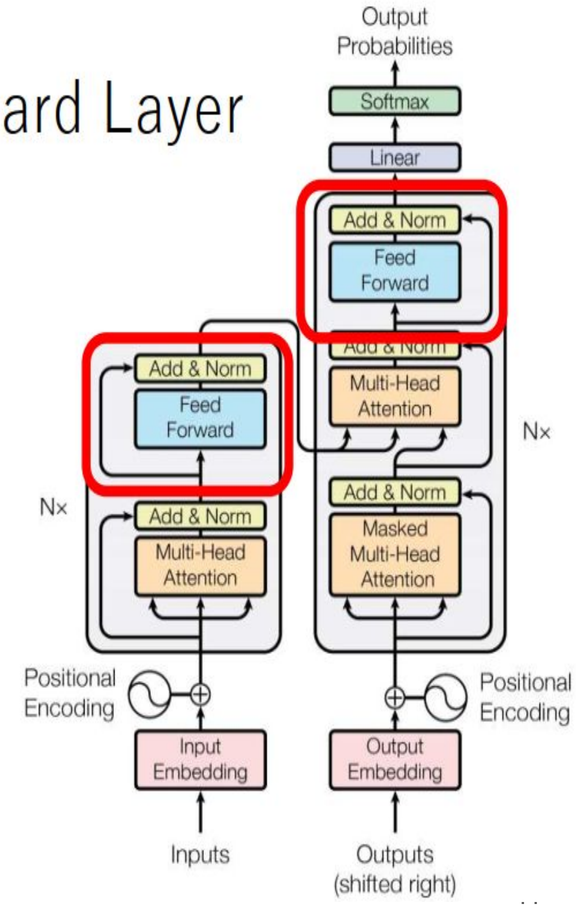


Entire Model Architecture

Attentions



Feed Forward Layer

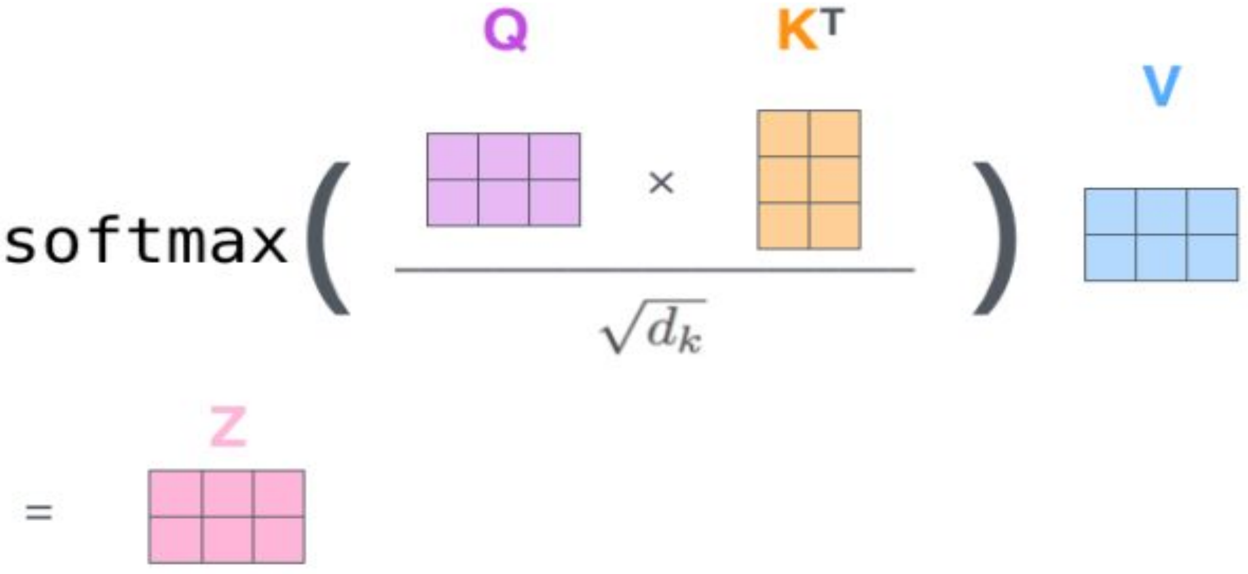


The encoder processes each item in the input sequence, it compiles the information it captures into a vector (called as context vector). The top encoder sends the context to all decoders, which begins producing the output sequence item by item.

What is Attention?

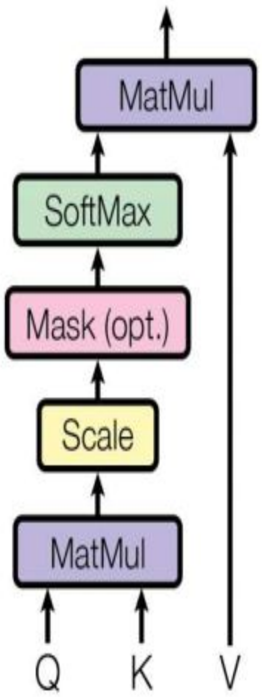
Attention : (vector, matrix, matrix) →vector

$$Attention(query, Key, Value) = Softmax(query \cdot Key^T) \cdot Value$$



The self-attention calculation in matrix form

Scaled Dot-Product Attention



- **Attention** maps a given query, key, value to a probability distribution, where the query, keys, values are 64 dimensional vectors. **Softmax function** is used **for multiclass classification as in example work "it" check different words with different probability values. And softmax gives**

Why Self Attention

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

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)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

- **n** = number of words in a longest sentence in the training set ~ 70
- **d** = number of hidden layers in neural network ~ 1000
- **Sequential operations** = amount of parallel computations
- **Maximum path length** = longest dependency within the sentence

(self attention: n^2 *d) 70 * 70 * 1000 << 70 * 1000 * 1000 (RNN or CNN)

Training the model: Experiment

- Data
  - WMT2014 English-German : 4.5 million sentence pairs
  - WMT2014 English-French : 36 million sentences
- Hardware and Schedule
  - 8 NVIDIA P100 GPUs
  - Base model : 100,000 steps or 12 hours
  - Big model : 300,000 steps (3.5 days)
- Optimizer : Adam
  - Warm up, and then decrease learning rate.

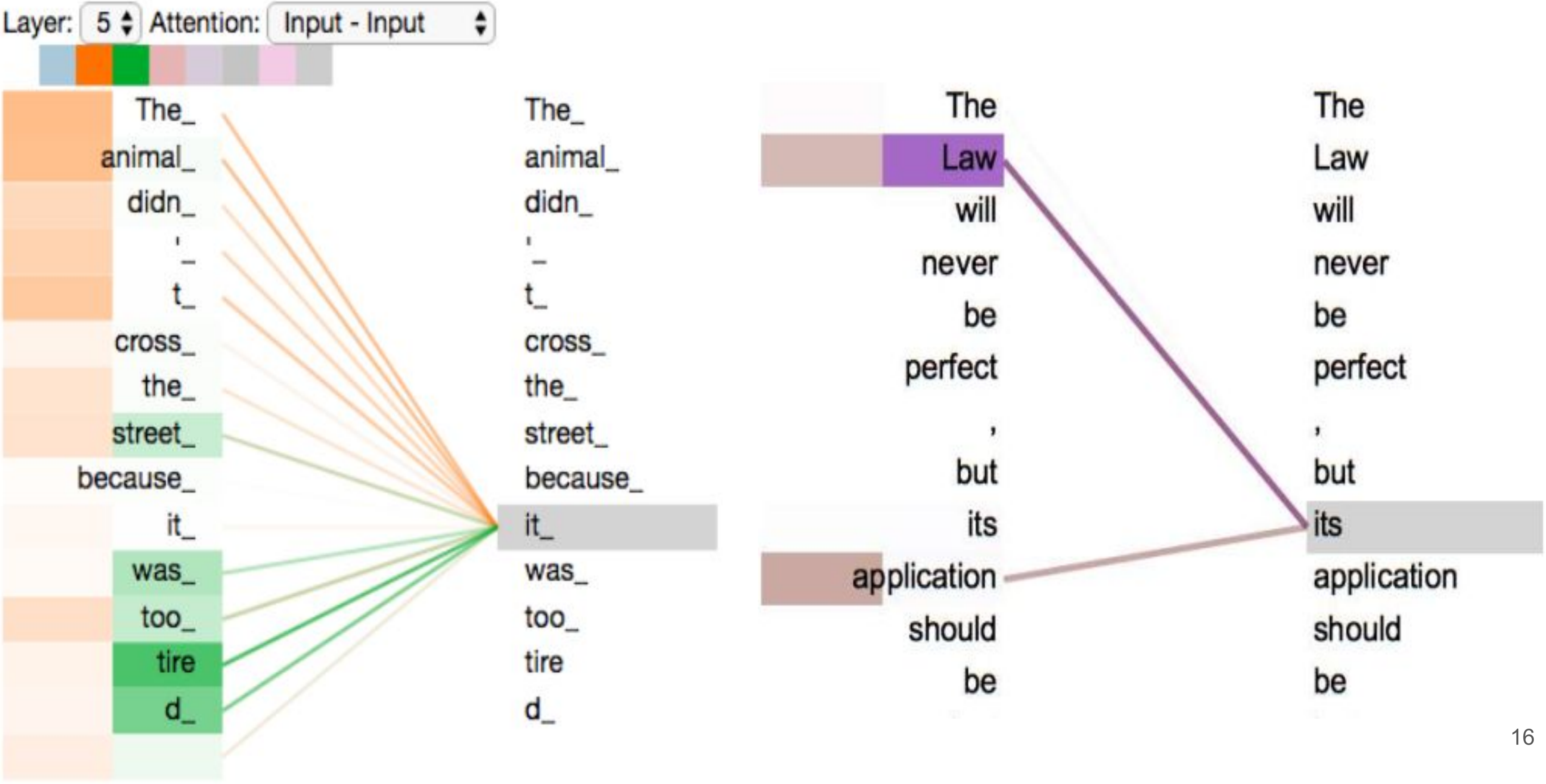
- Trained a 4-layer (4 encoders-decoders) 1024 dimensional transformer model on the **Wall Street Journal (WSJ)** portion about 40K training sentences. They have also trained on **BerkleyParser** corpora from with approximately 17M sentences.

Experiment - Result

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

Self-attention visualization



Summary

- I talked about a novel method "***Transformer***" - the first sequence to sequence machine translation process entirely based on ***Attention***, replacing recurrent layers in neural network with ***multi-headed self-attention***.

- **Future work** on impactful applications:
    - Current range of **self attention neighbourhood = r (fixed)** in input sequence that is centered around a respective output position. They will look for different values of r for dependencies in a given sentence.
    - To sum up reading comprehension.
    - Abstractive summarization of research papers and news articles.

Thank you for your

Attention

Backup: Methodology (matrix multiplications)

X = **512** dimensional matrix (vector embedding)

Q = **K**= **V**= **dk** = **64** dimensional matrices

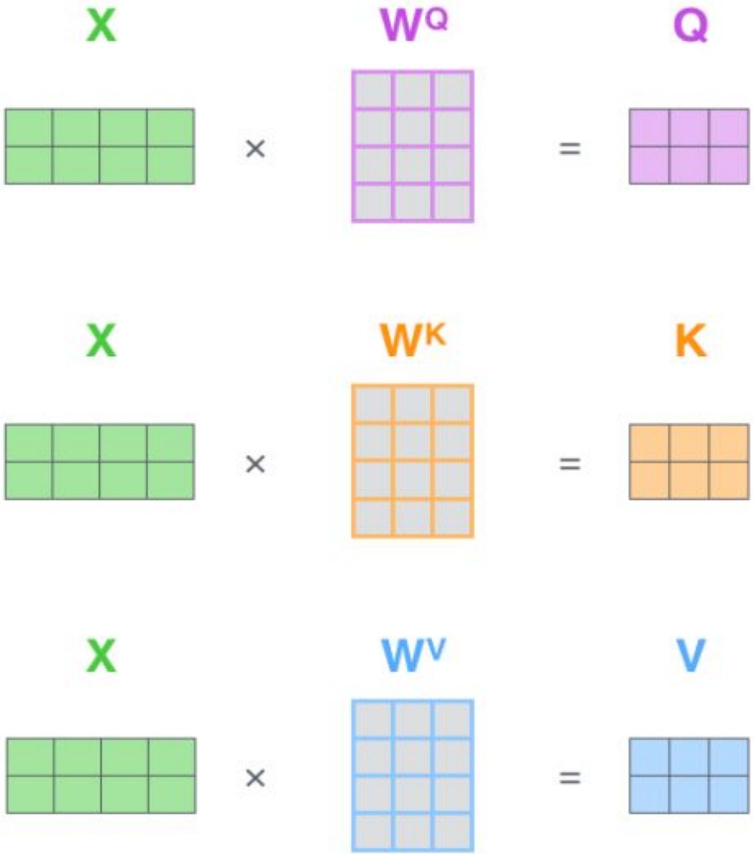
Trainable matrices **WQ, WK, WV** are used for **8** matrix multiplications in case of **multi-headed attention**

X1 is multiplied by **WQ weight matrix** produce **q1 query vector** for that word

X2 is multiplied by **WQ weight matrix** produce **q2 query vector** for 2nd word and so on

Matrix **W0** is used for **1** matrix multiplication

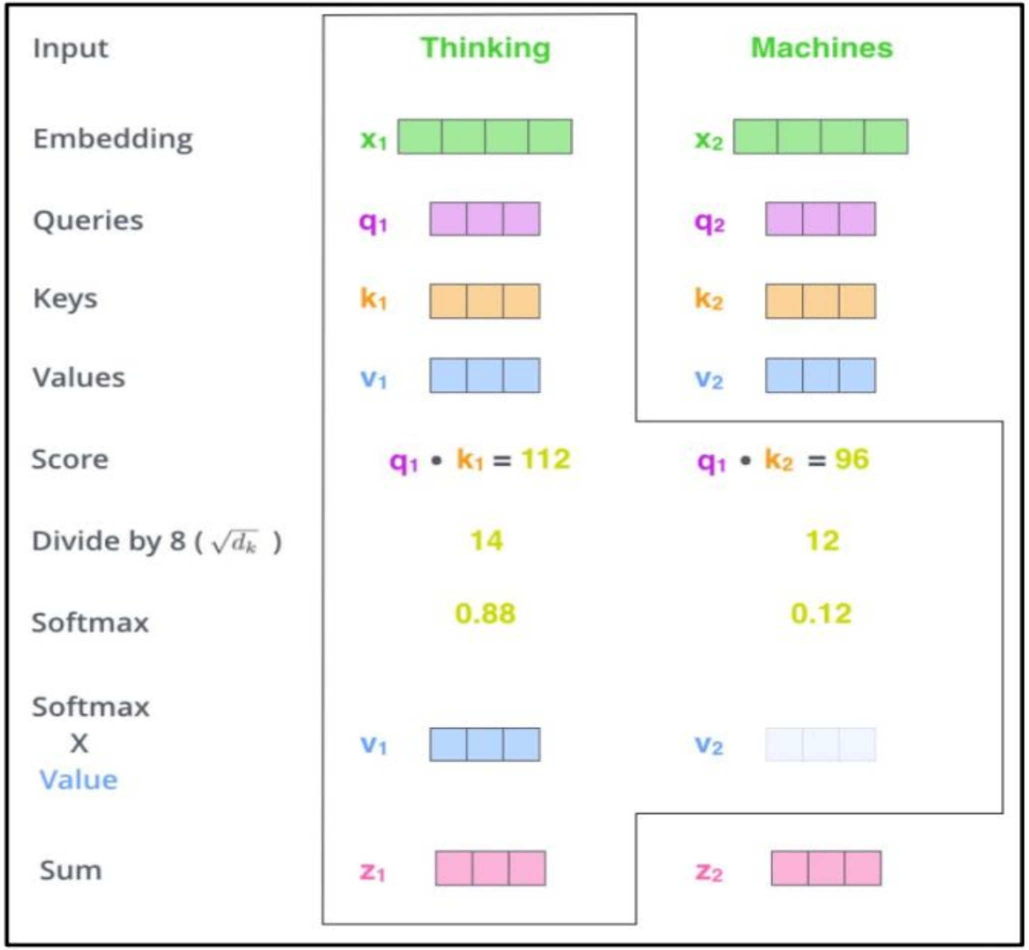
There are 8 **Attention heads** (Z0 ... Z7 = Zi)



Backup

Self Attention

1. Find q, k and v
2. Find $score_i = q_i * k_j$
3. Normalise
4. Softmax
5. Multiply softmax output with v
6. Sum weighted value vector



Backup

Experiment – evaluation metrics

- **BLEU** (Bilingual Evaluation Understudy)
  - Evaluation metrics on how machine translation(MT) and ground truth(GT) translation are similar.

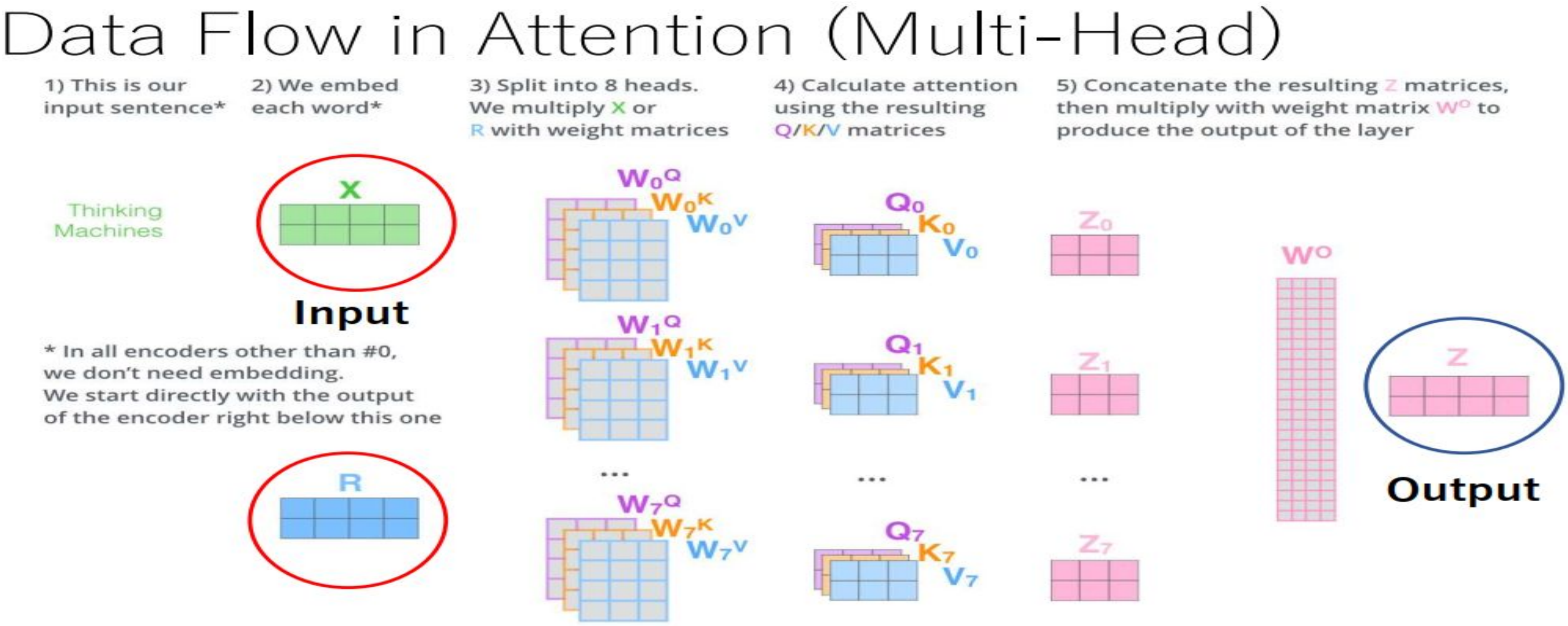
$$BLEU = BP_{BLEU} \cdot \exp\left(\sum_{n=1}^{N} \frac{1}{N} log p_n\right)$$

  Usually, N=4.

  - BP_{BLEU} : penalty if multiplied when len(MT) < len(GT)

  - $p_n = \frac{\sum_i the\ number\ of\ n\text{−}grams\ same\ in\ MT_i\ and\ GT_i}{\sum_i the\ number\ of\ n\text{−}grams\ \ in\ MT_i}$

Backup



- **Multihead (8 headed) will increase representational power(multiple Zi at a time)** as single attention head(Z1) could focus only on first word.
- **Concatenate all attention heads (Z0 ... Z7) = Z * W0 = Final Z**