자연어 처리 DAY 4
Transformer

Jaegul Choo
Associate Professor, Graduate School of AI, KAIST

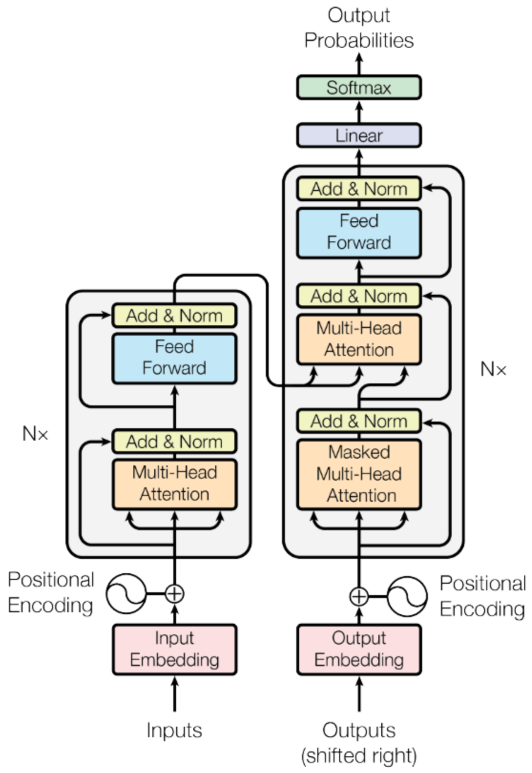
boostcamp AI Tech

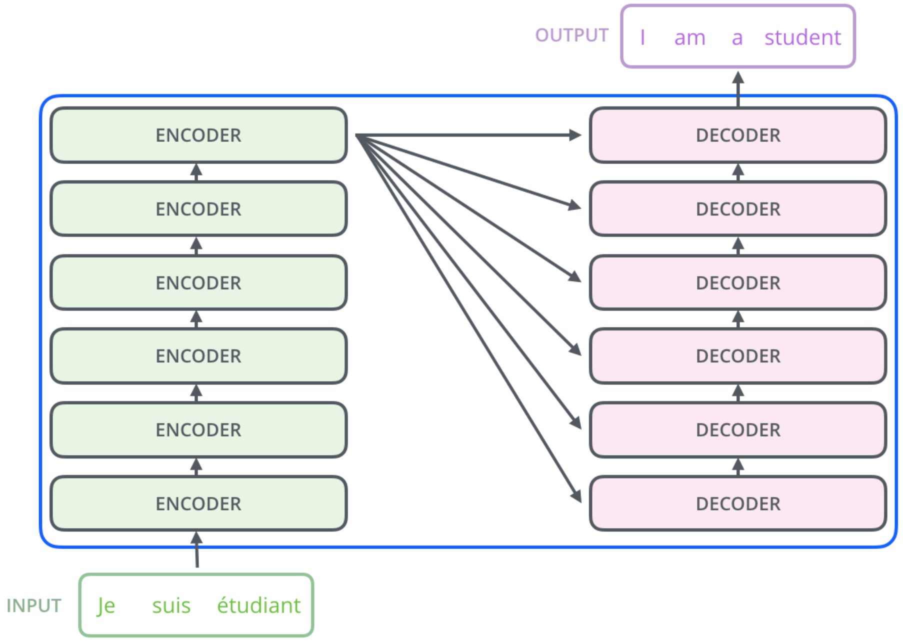
DAVIAN
Data and Visual Analytics Lab

1.
Transformer

Transformer: High-Level View

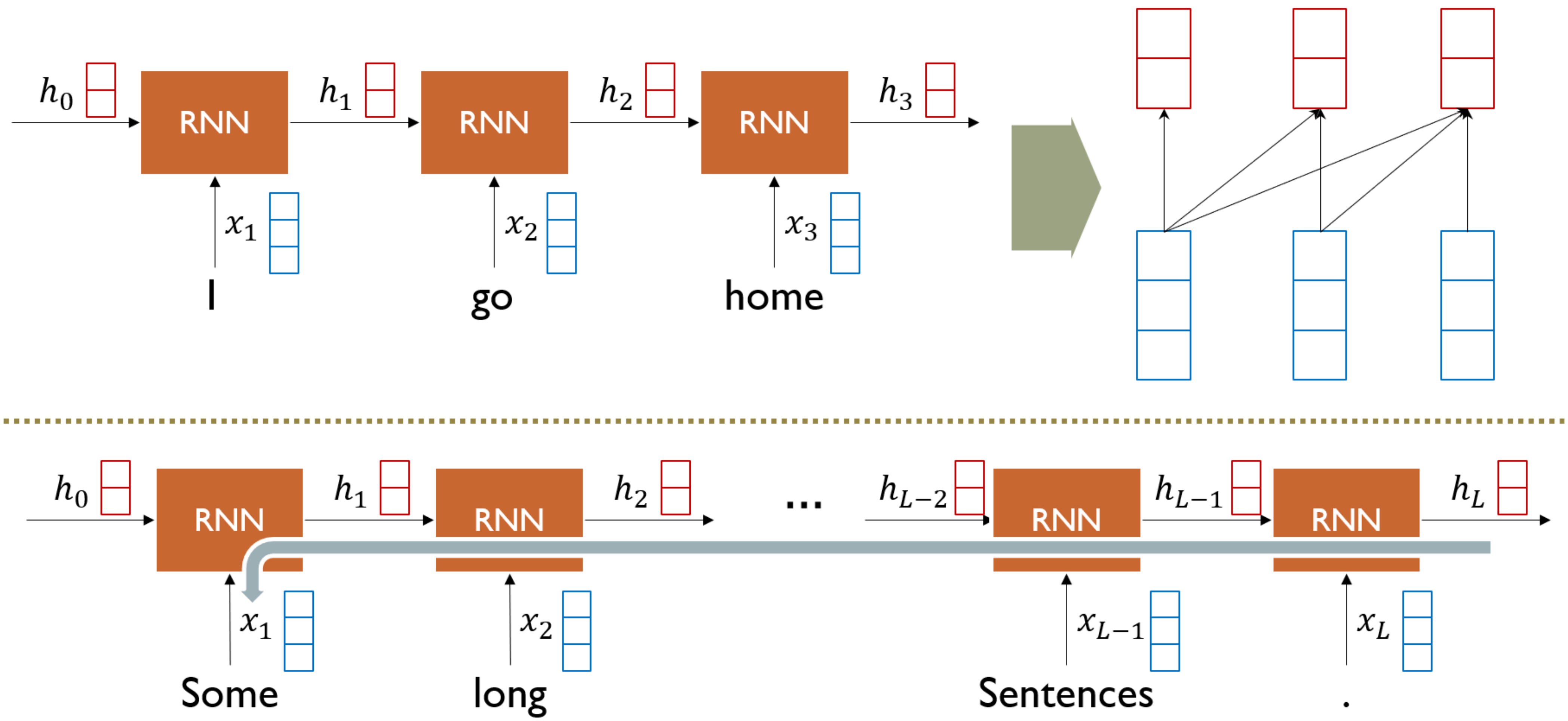
- **Attention is all you need, NeurIPS'17**
  - No more RNN or CNN modules



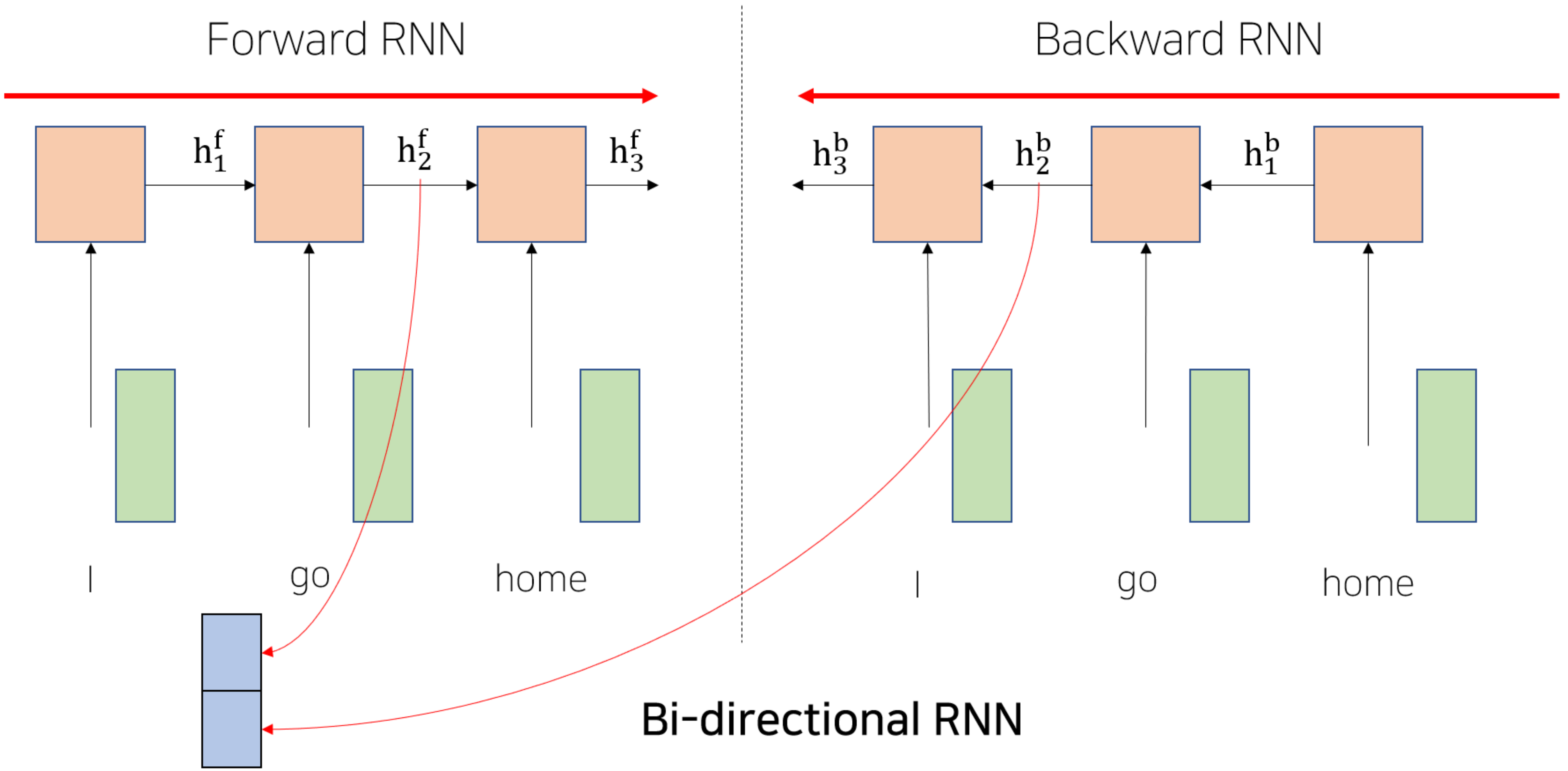


Attention Is All You Need, NeurIPS'17
http://jalammar.github.io/illustrated-transformer/

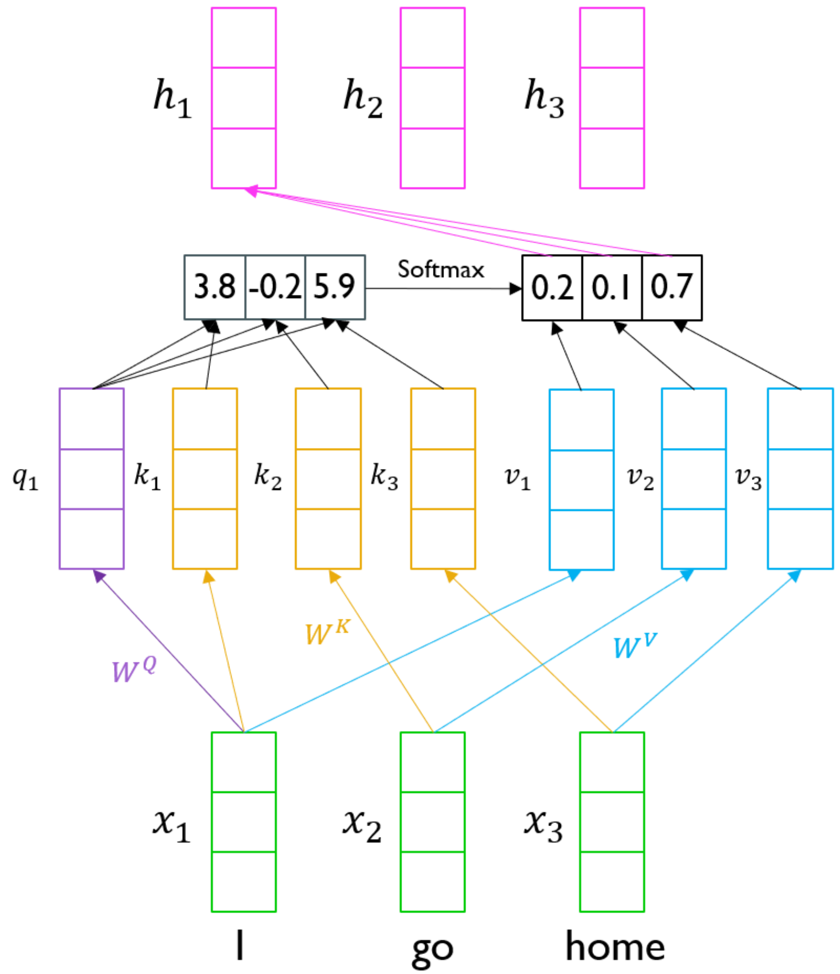
RNN: Long-Term Dependency

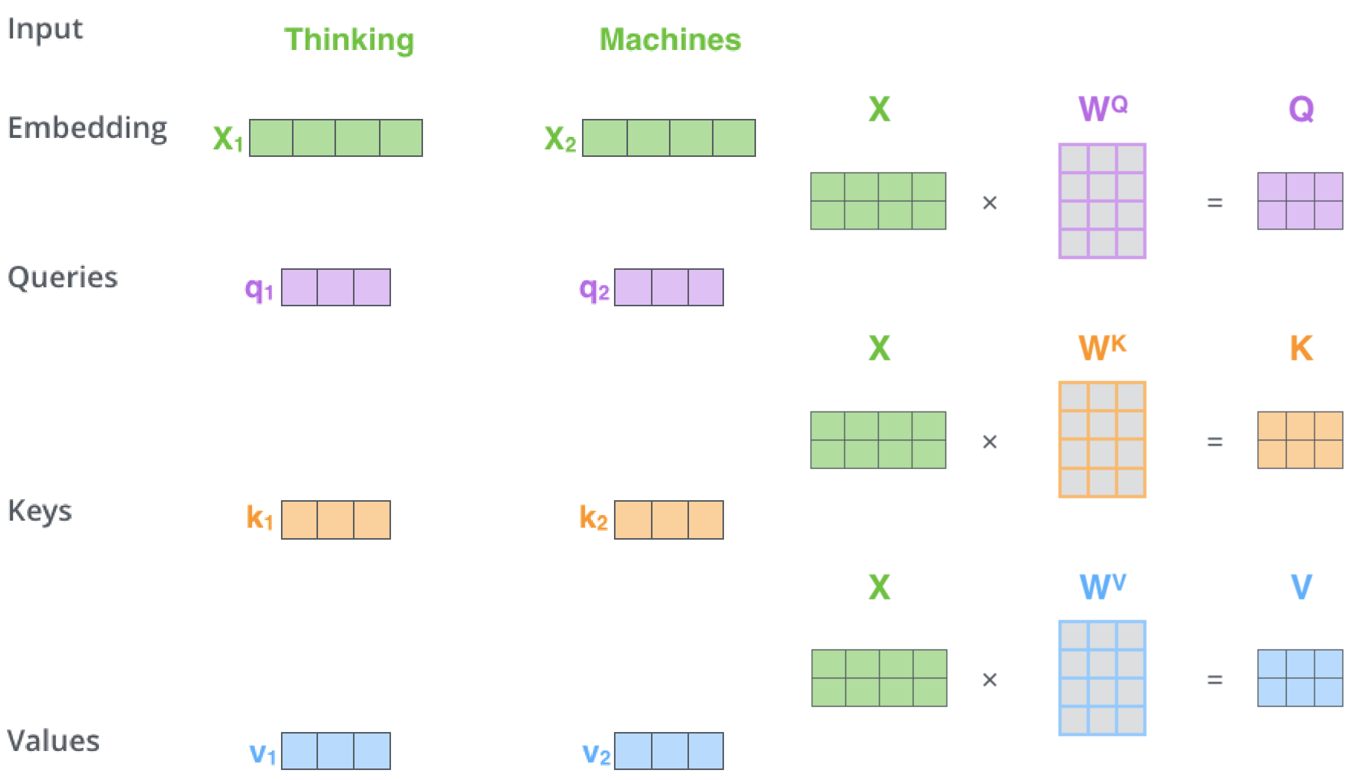


Bi-Directional RNNs



Transformer: Long-Term Dependency





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

Transformer: Scaled Dot-Product Attention

- Inputs: a query $\boldsymbol{q}$ and a set of key-value $(\boldsymbol{k}, \boldsymbol{v})$ pairs to an output
- Query, key, value, and output is all vectors

- Output is weighted sum of values
- Weight of each value is computed by an inner product of query and corresponding key
- Queries and keys have same dimensionality $\boldsymbol{d_k}$, and dimensionality of value is $\boldsymbol{d_v}$

$$A(q, K, V) = \sum_i \frac{\exp(q \cdot k_i)}{\sum_j \exp(q \cdot k_j)} v_i$$

Transformer: Scaled Dot-Product Attention

- When we have multiple queries $\boldsymbol{q}$, we can stack them in a matrix $\boldsymbol{Q}$:

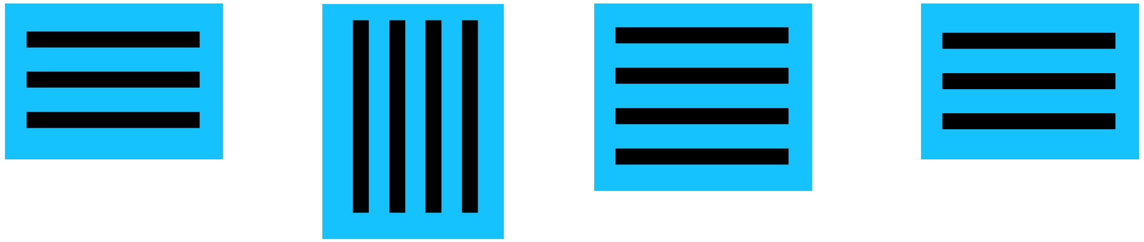
$$A(q, K, V) = \sum_i \frac{\exp(q \cdot k_i)}{\sum_j \exp(q \cdot k_j)} v_i$$

- Becomes:

$$A(Q, K, V) = \mathrm{softmax}(QK^T)V$$

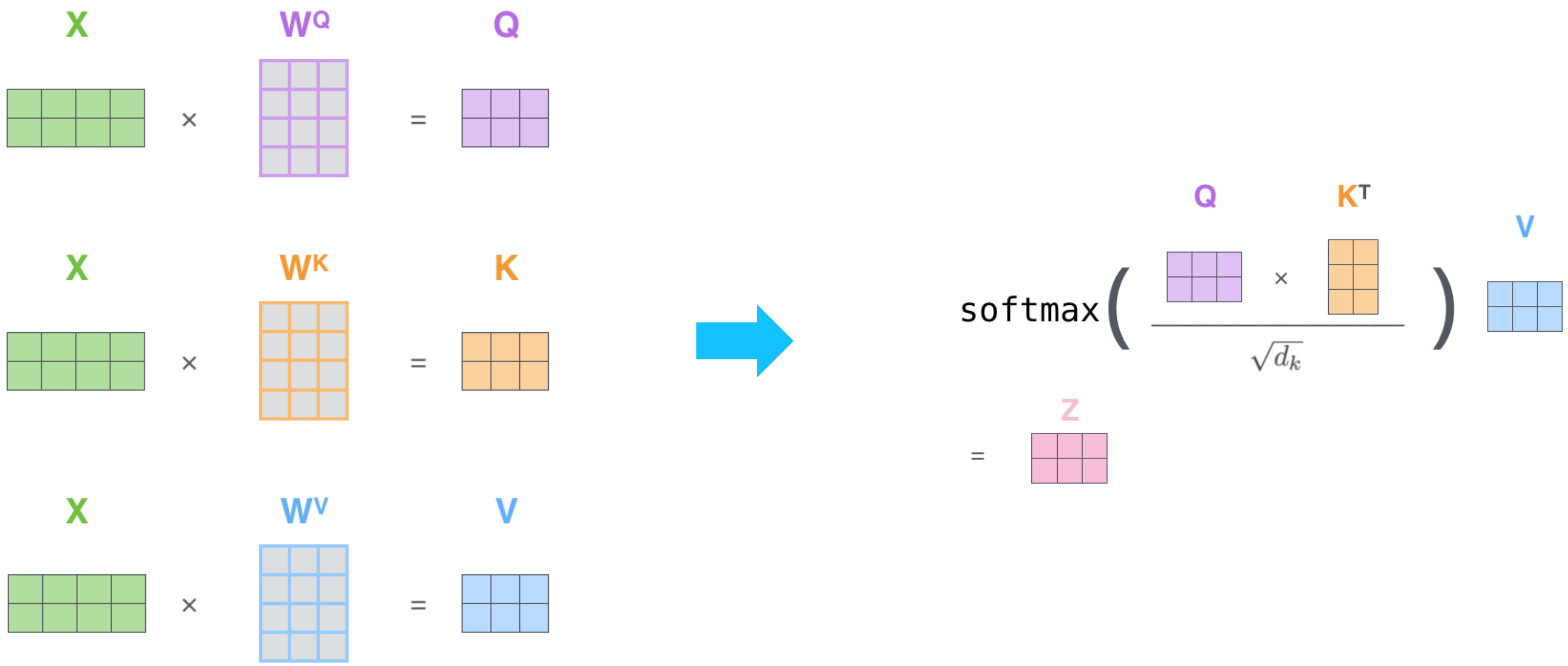
$$(|Q| \times d_k) \times (d_k \times |K|) \times (|V| \times d_v) = (|Q| \times d_v)$$

Row-wise softmax



Transformer: Scaled Dot-Product Attention

- Example from illustrated transformer



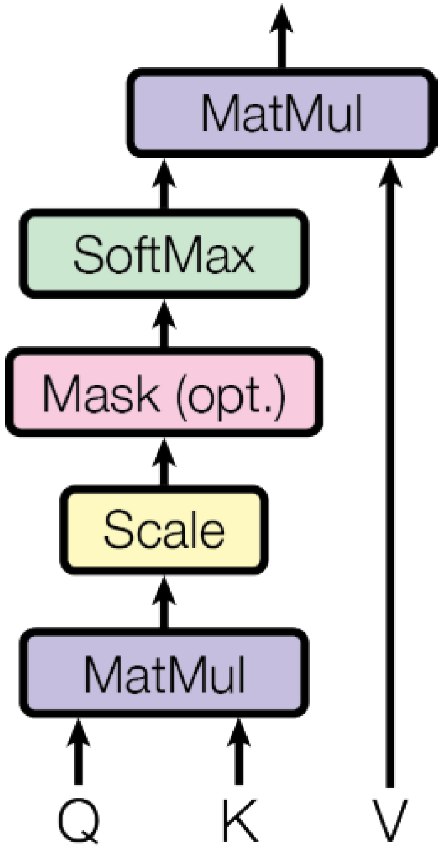
http://jalammar.github.io/illustrated-transformer/

Transformer: Scaled Dot-Product Attention

- **Problem**
  - As d_k gets large, the variance of $q^T k$ increases
  - Some values inside the softmax get large
  - The softmax gets very peaked
  - Hence, its gradient gets smaller

- **Solution**
  - Scaled by the length of query / key vectors:

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



Attention Is All You Need, NeurIPS'17

2.

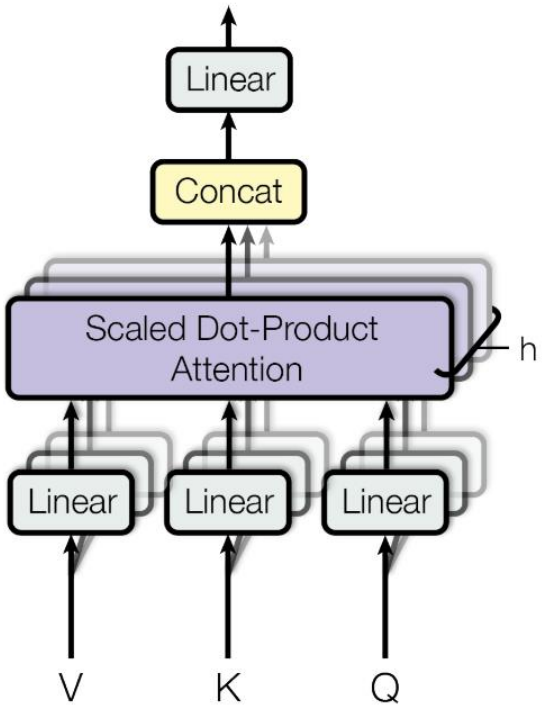
Transformer (cont'd)

Transformer: Multi-Head Attention

- The input word vectors are the queries, keys and values
- In other words, the word vectors themselves select each other
- **Problem of single attention**
  - Only one way for words to interact with one another
- **Solution**
  - Multi-head attention maps Q, K, V into

    the h number of lower-dimensional spaces via W matrices
- Then apply attention, then concatenate outputs

  and pipe through linear layer



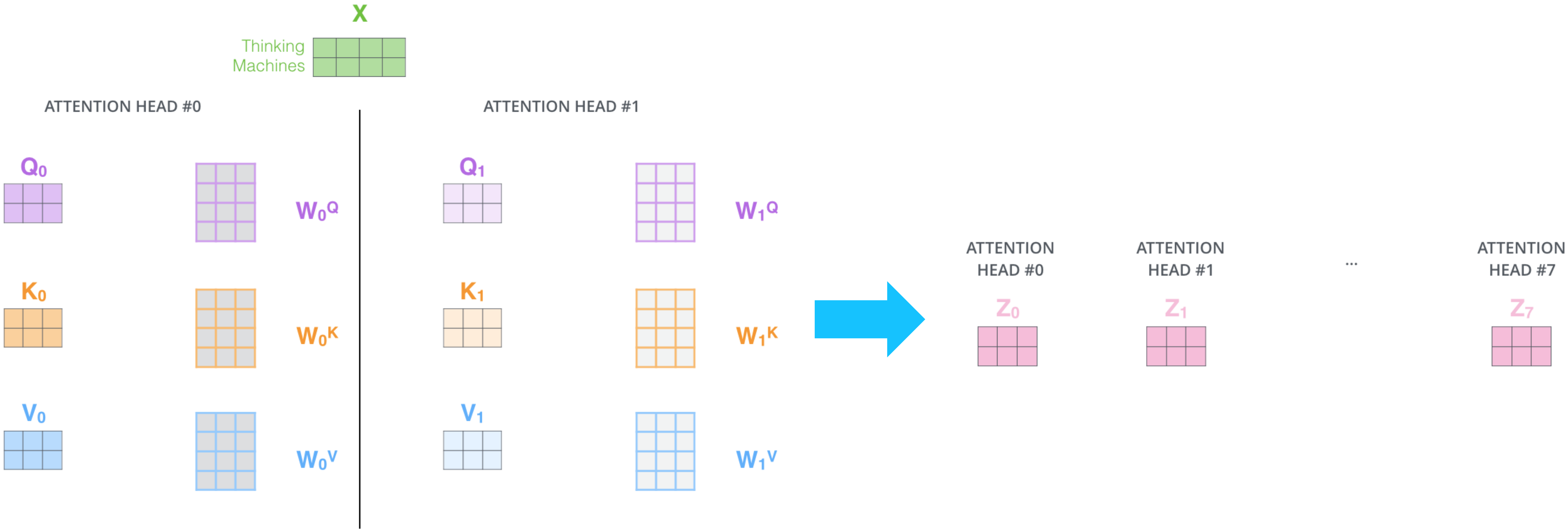
$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)W^O$$

$$\text{where } \mathrm{head}_i = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Attention Is All You Need, NeurIPS'17

Transformer: Multi-Head Attention

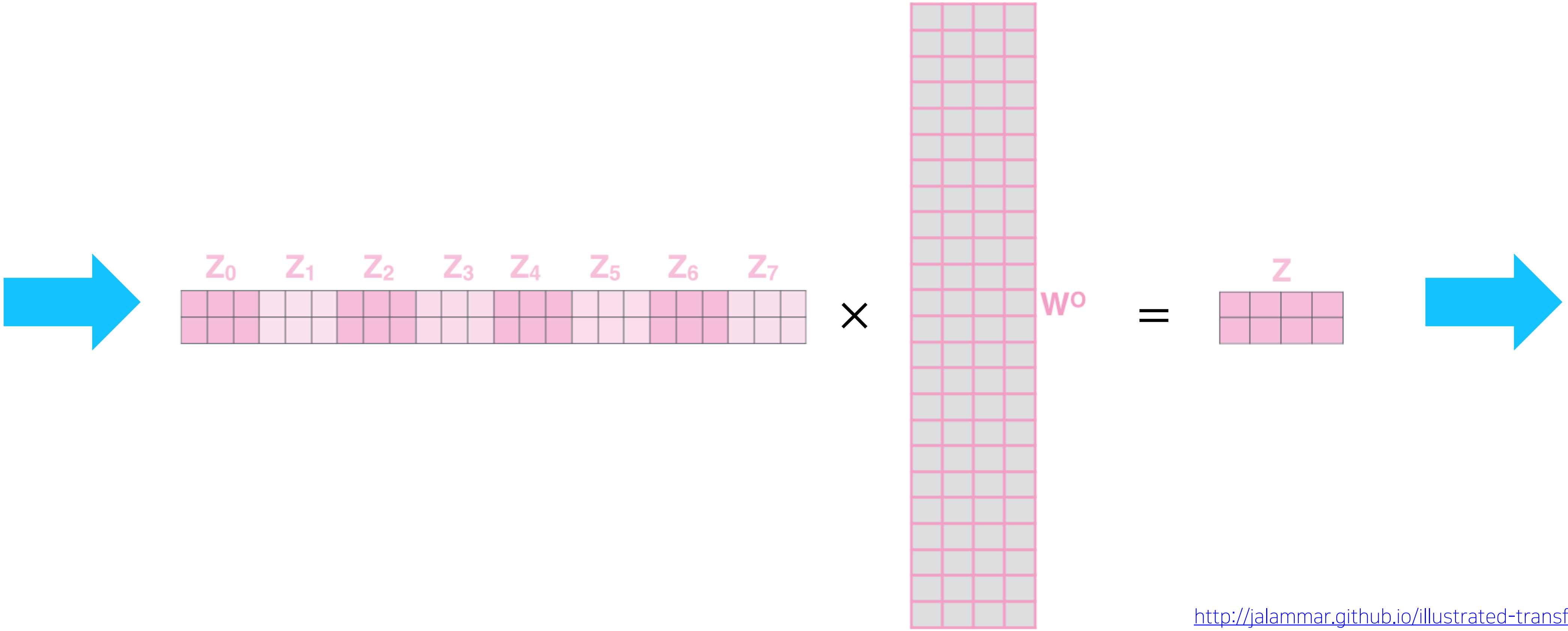
- Example from illustrated transformer



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

Transformer: Multi-Head Attention

- Example from illustrated transformer



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

Transformer: Multi-Head Attention

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

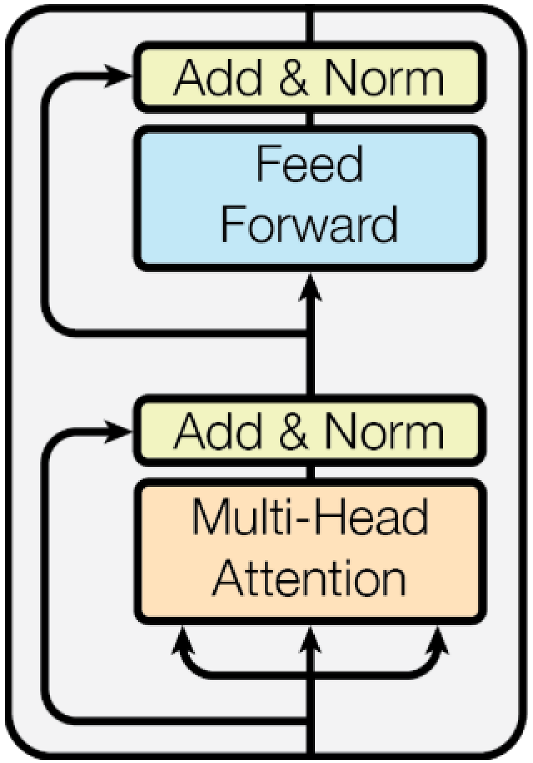
Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Attention Is All You Need, NeurIPS'17

Transformer: Block-Based Model

- **Each block has two sub-layers**
  - Multi-head attention
  - Two-layer feed-forward NN (with ReLU)

- **Each of these two steps also has**
  - Residual connection and layer normalization:
  - $LayerNorm(x + sublayer(x))$

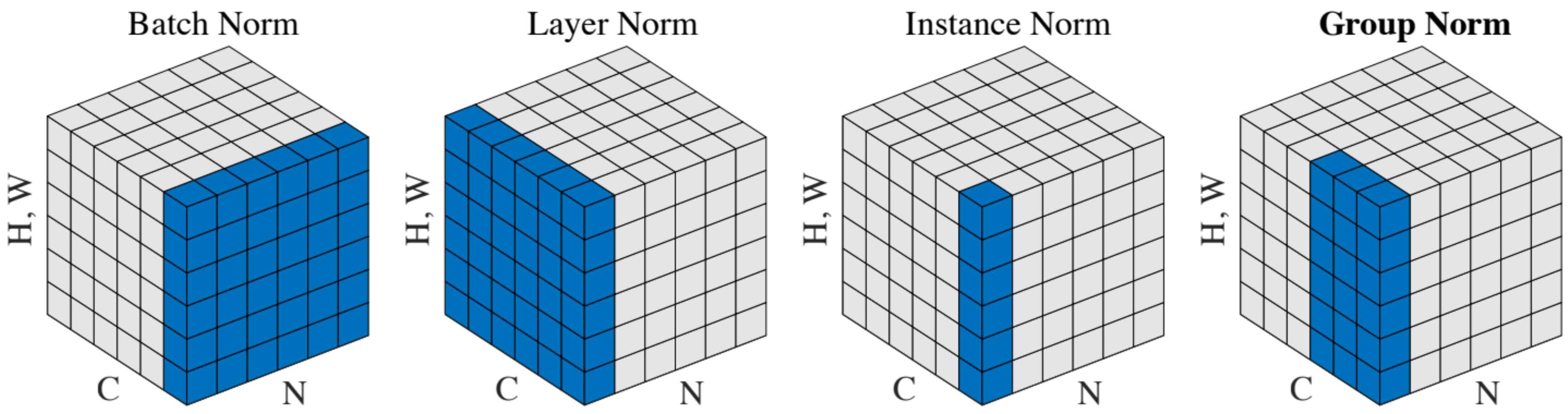


Attention Is All You Need, NeurIPS'17

Transformer: Layer Normalization

- Layer normalization changes input to have zero mean and unit variance, per layer and per training point (and adds two more parameters)

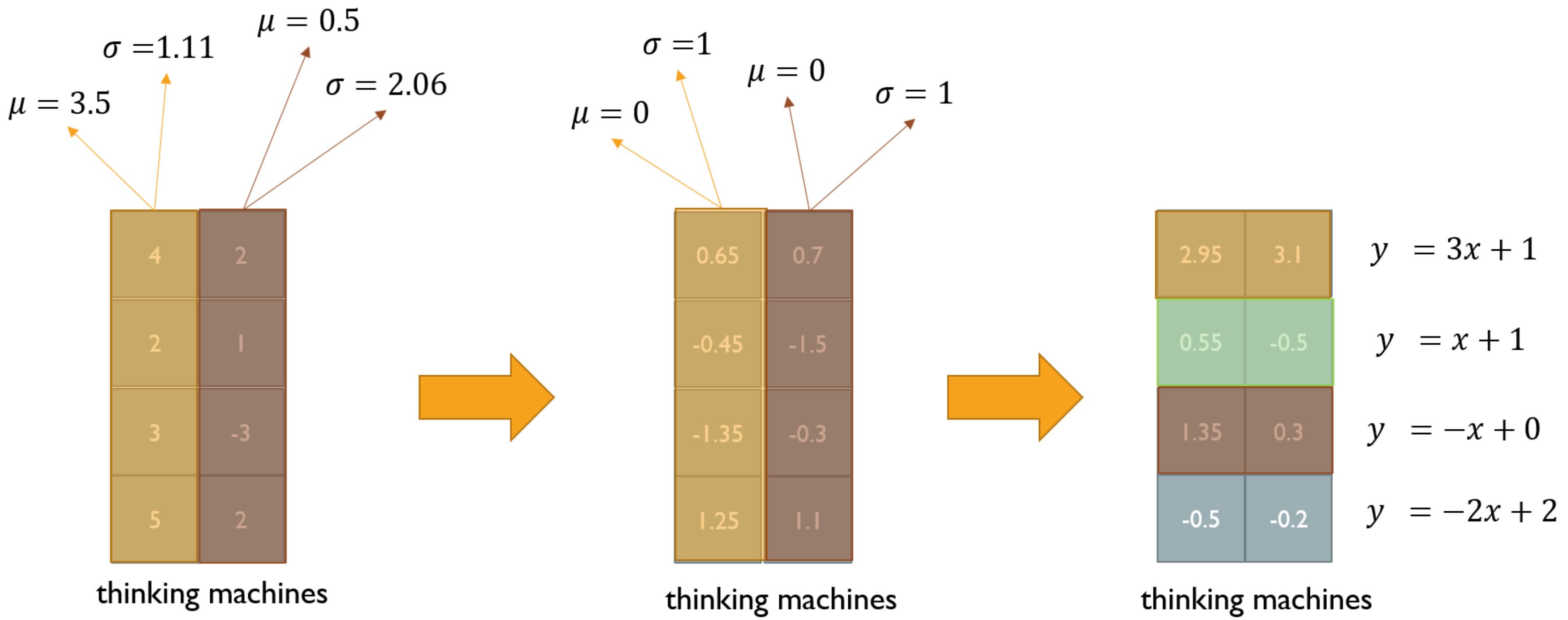
$$\mu^{l} = \frac{1}{H}\sum_{i=1}^{H} a_i^l, \qquad \sigma^{l} = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(a_i^l - \mu^l)^2}, \qquad h_i = f\left(\frac{g_i}{\sigma_i}(a_i - \mu_i) + b_i\right)$$



Group Normalization, ECCV'18

Transformer: Layer Normalization

- **Layer normalization consists of two steps:**
  - Normalization of each word vectors to have mean of zero and variance of one.
  - Affine transformation of each sequence vector with learnable parameters



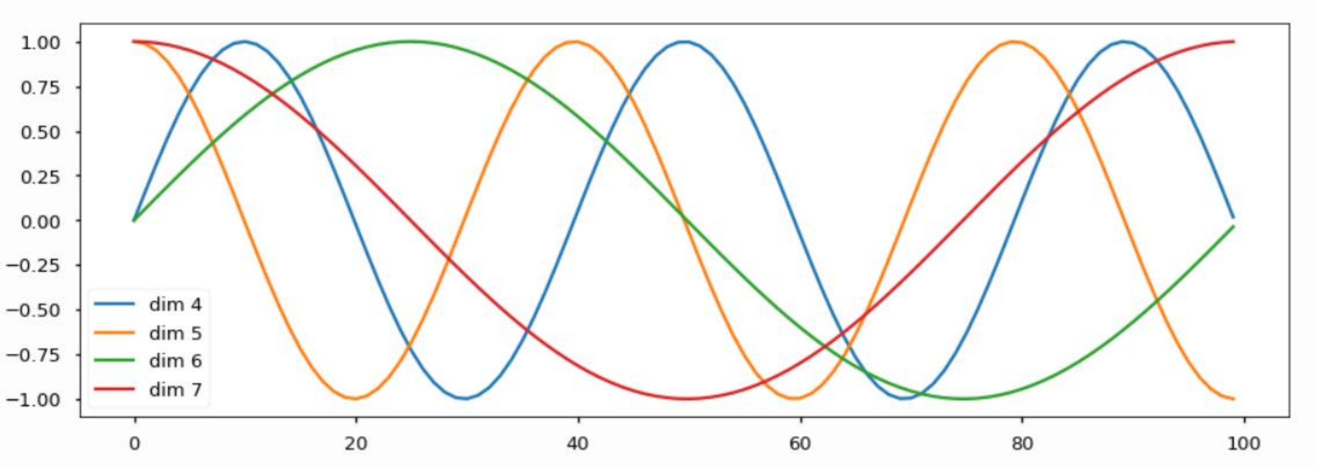
Transformer: Positional Encoding

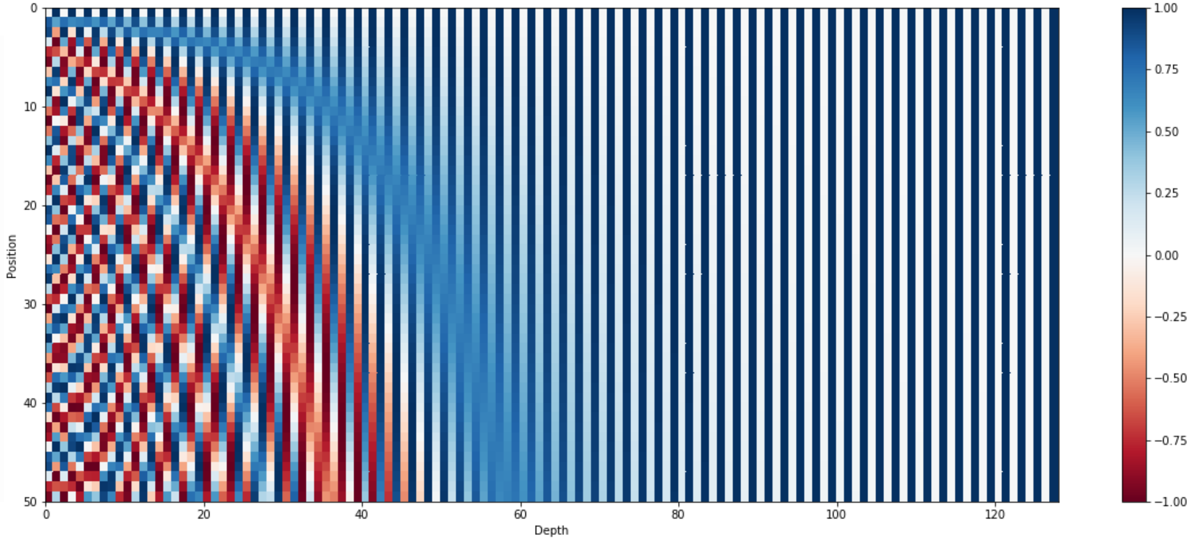
- Use sinusoidal functions of different frequencies

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

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

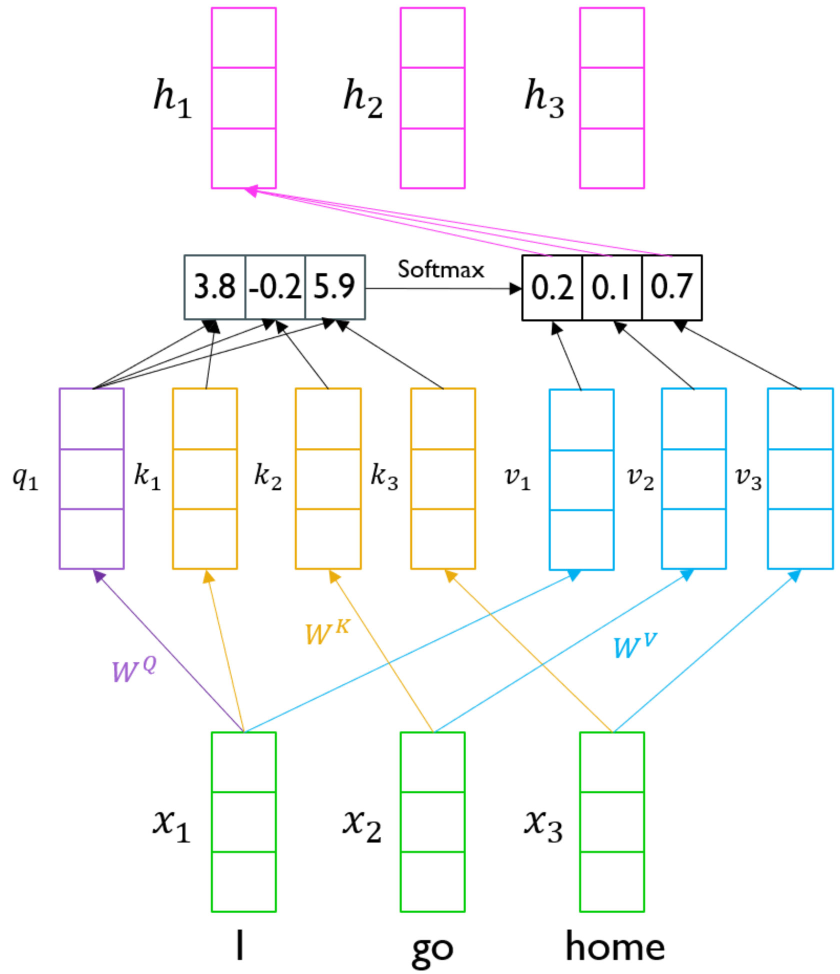
- Easily learn to attend by relative position, since for any fixed offset k, $PE_{(pos+k)}$ can be represented as linear function of $PE_{(pos)}$

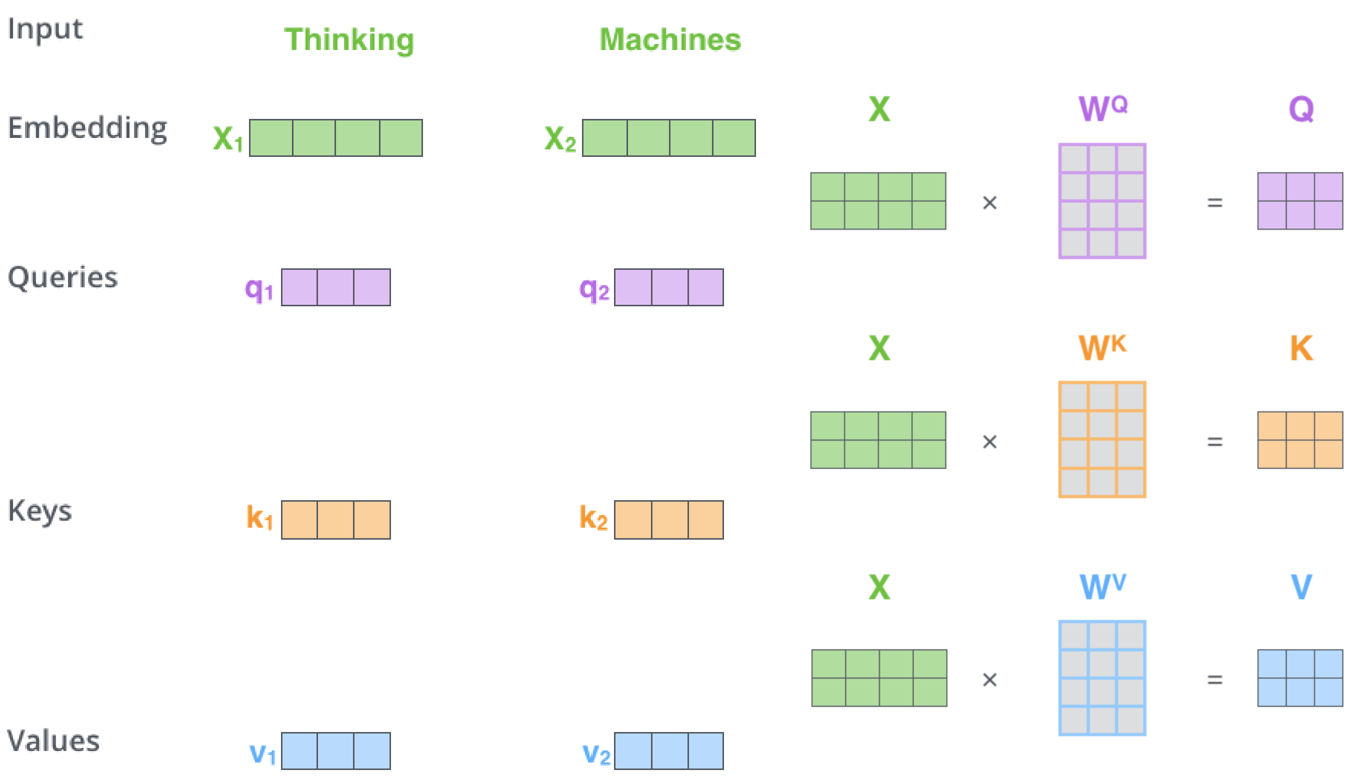




http://nlp.seas.harvard.edu/2018/04/03/attention

Transformer: Long-Term Dependency

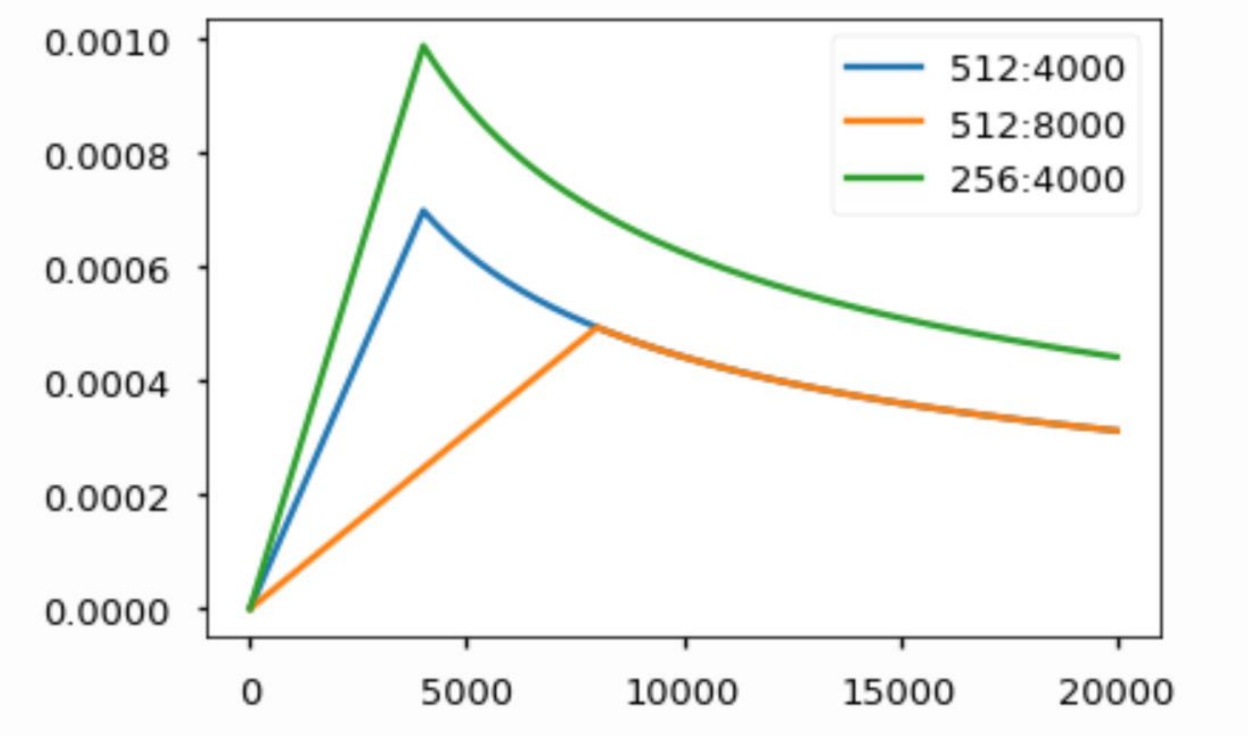




http://jalammar.github.io/illustrated-transformer/

Transformer: Warm-up Learning Rate Scheduler

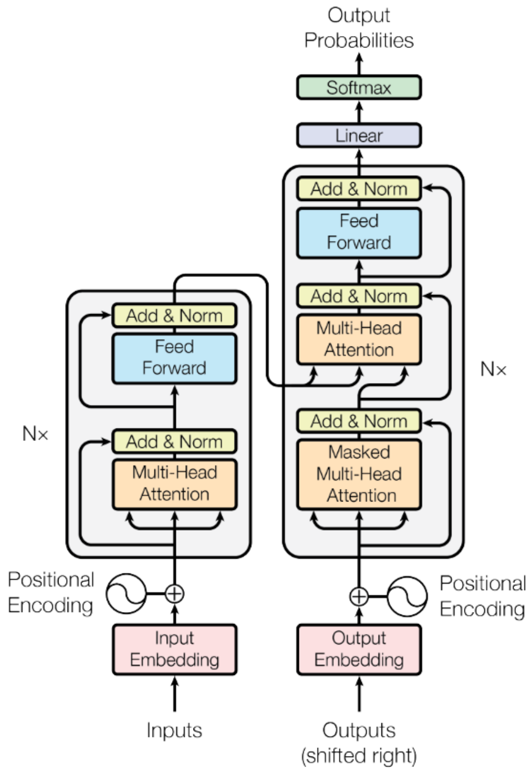
- $learning\ rate = d_{model}^{-0.5} \cdot \min(\#step^{-0.5}, \#step \cdot warmup\_steps^{-1.5})$

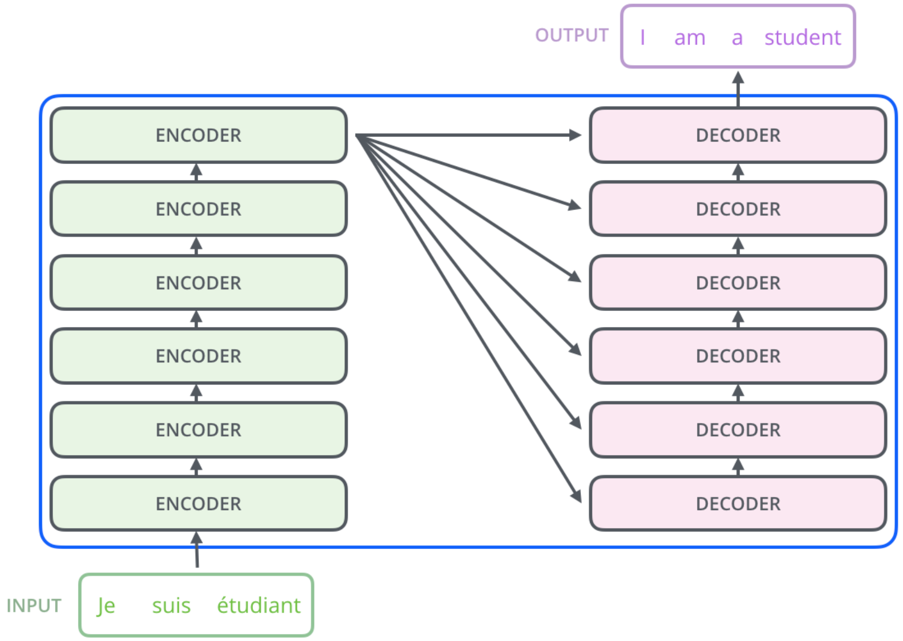


Group Normalization, ECCV'18

Transformer: High-Level View

- **Attention is all you need, NeurIPS'17**
  - No more RNN or CNN modules

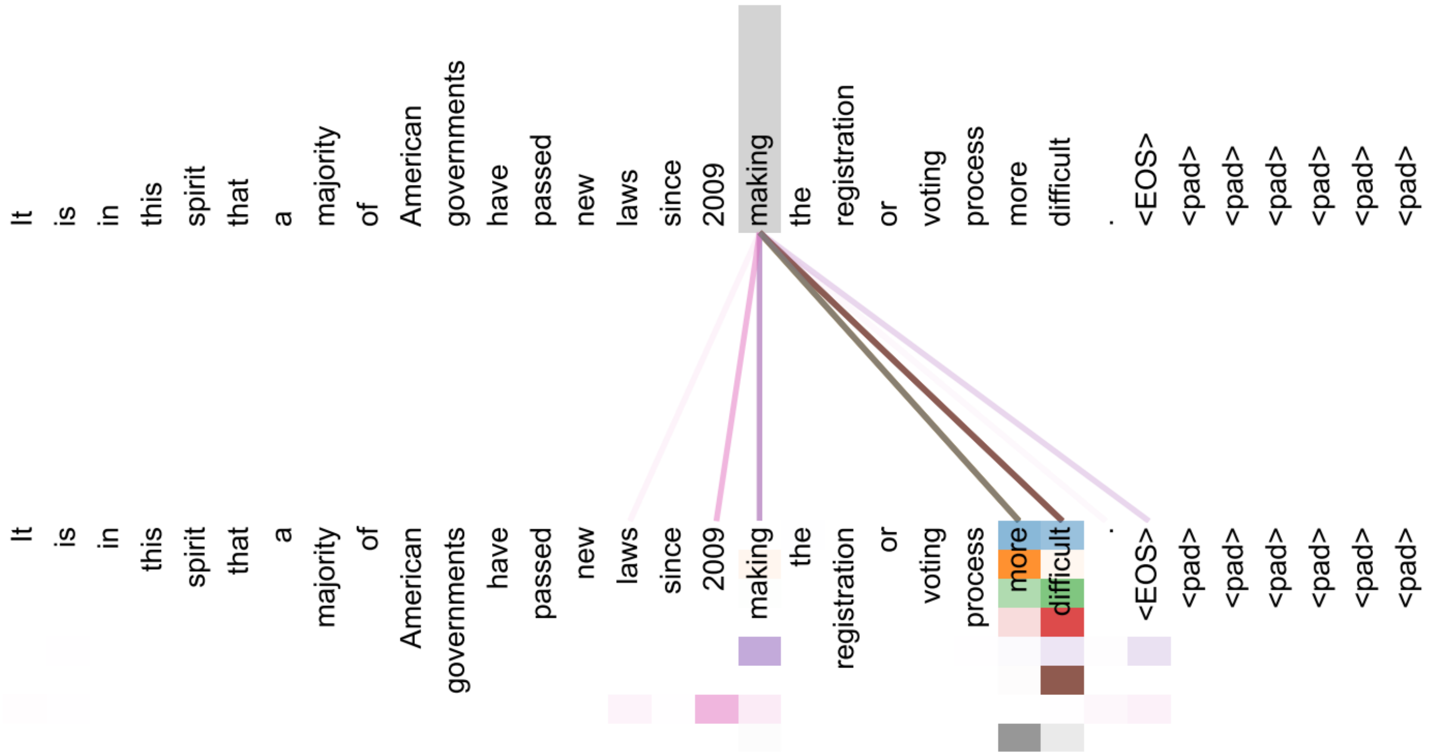




Attention Is All You Need, NeurIPS'17
http://jalammar.github.io/illustrated-transformer/

Transformer: Encoder Self-Attention Visualization

- Words start to pay attention to other words in sensible ways

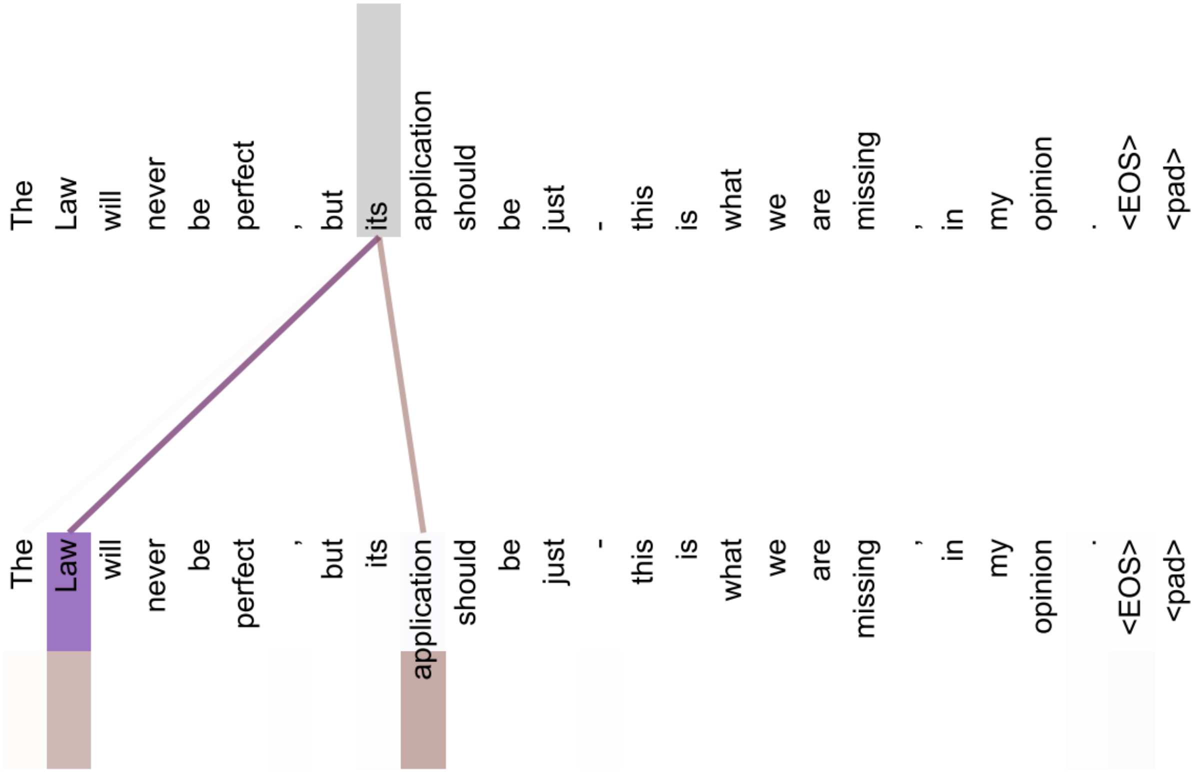


https://colab.research.google.com/github/tensorflow/tensor2tensor/blob/master/tensor2tensor/notebooks/hello_t2t.ipynb

Attention Is All You Need, NeurIPS'17

Transformer: Encoder Self-Attention Visualization

- Words start to pay attention to other words in sensible ways

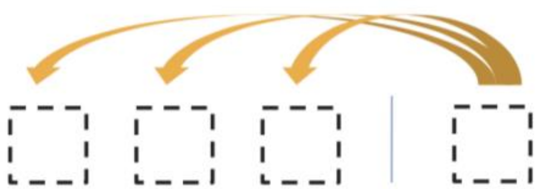


https://colab.research.google.com/github/tensorflow/tensor2tensor/blob/master/tensor2tensor/notebooks/hello_t2t.ipynb

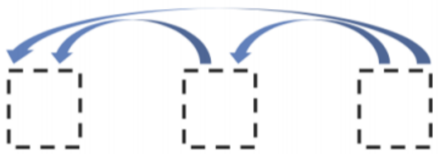
Attention Is All You Need, NeurIPS'17

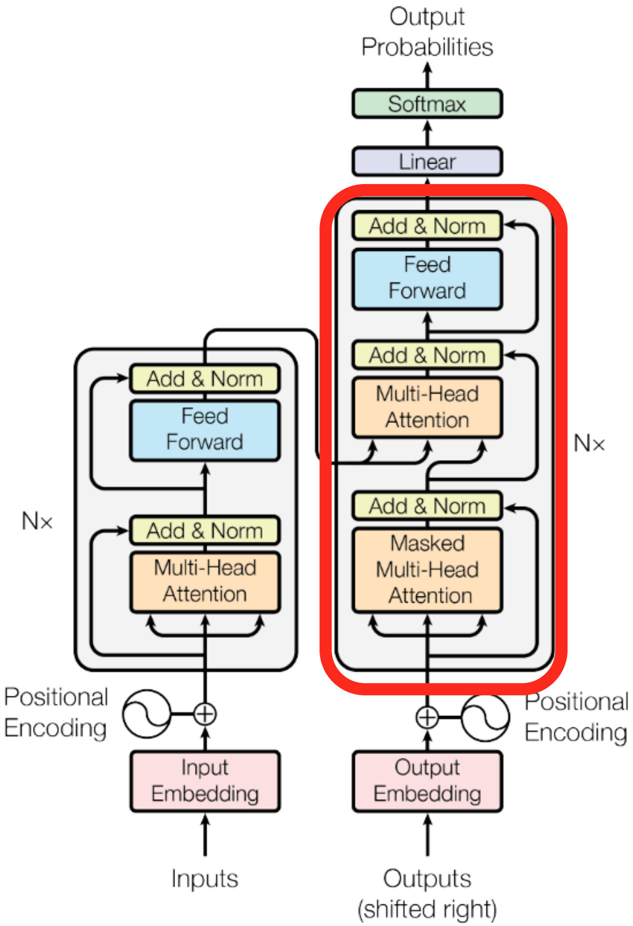
Transformer: Decoder

- Two sub-layer changes in decoder
- Masked decoder self-attention on previously generated output



- Encoder-Decoder attention,
where queries come from previous decoder layer
and keys and values come from output of encoder

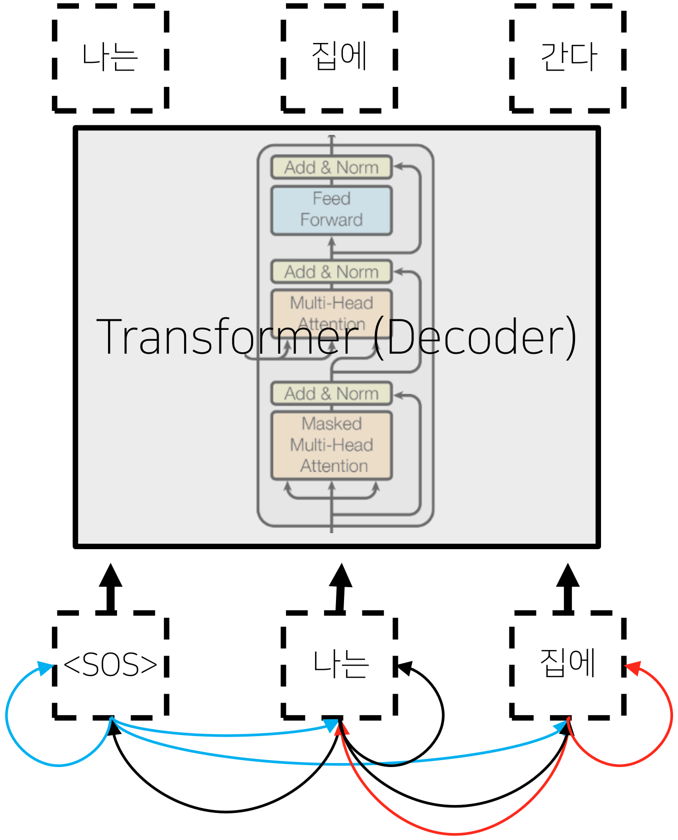


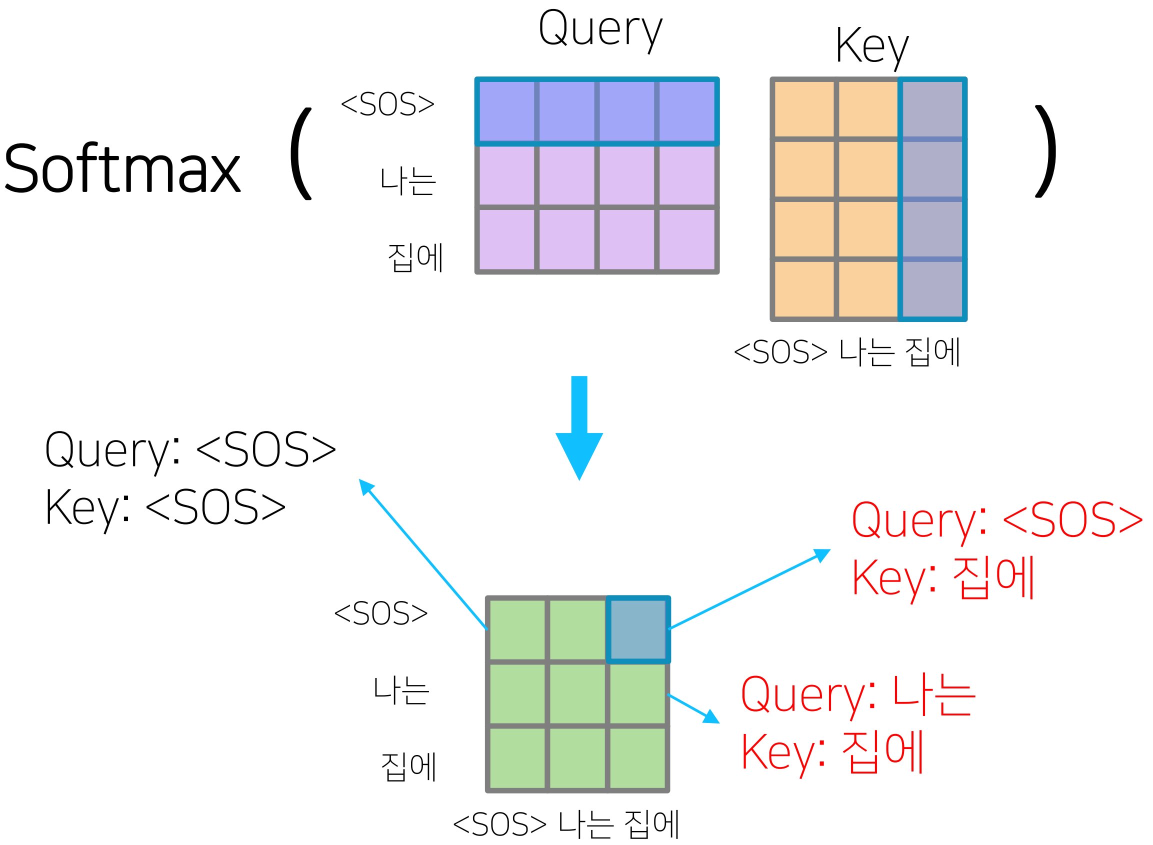


Attention Is All You Need, NeurIPS'17

Transformer: Masked Self-Attention

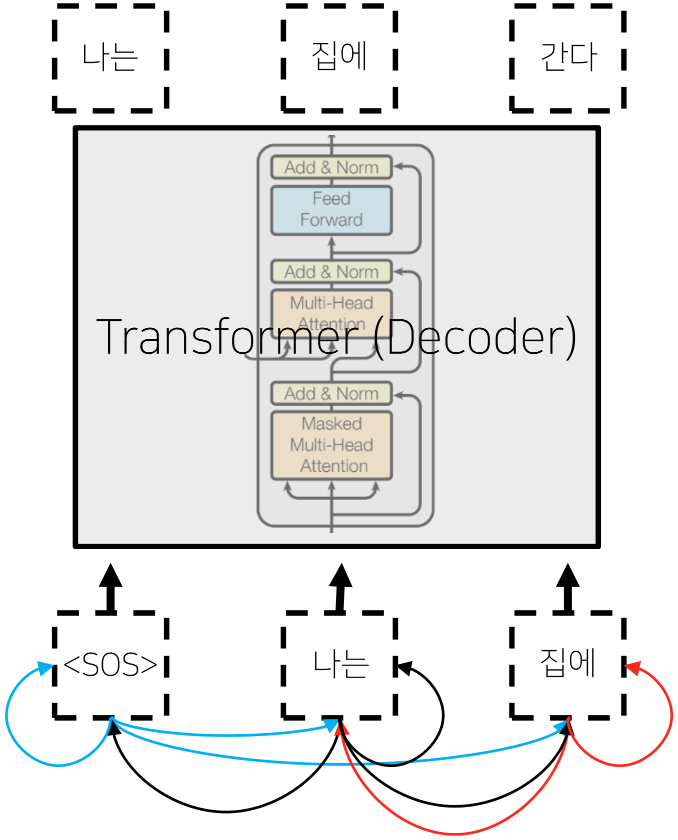
- Those words not yet generated cannot be accessed during the inference time
- Renormalization of softmax output prevents the model from accessing ungenerated words





Transformer: Masked Self-Attention

- Those words not yet generated cannot be accessed during the inference time
- Renormalization of softmax output prevents the model from accessing ungenerated words



Query: <SOS>
Key: <SOS>

Query: <SOS>
Key: 집에

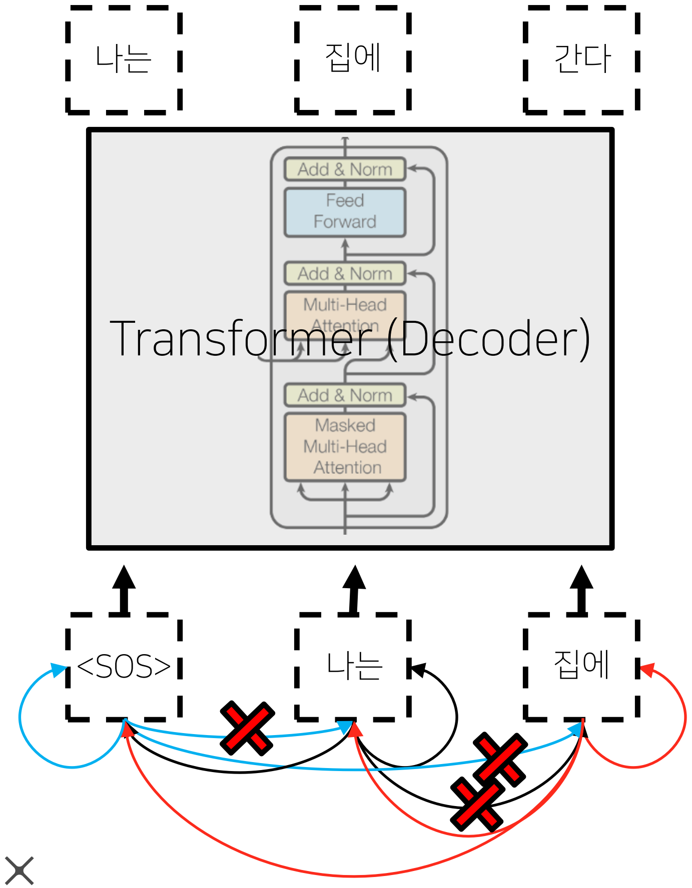
	<SOS>	나는	집에
<SOS>	0.91	0.05	0.04
나는	0.42	0.47	0.11
집에	0.25	0.31	0.44

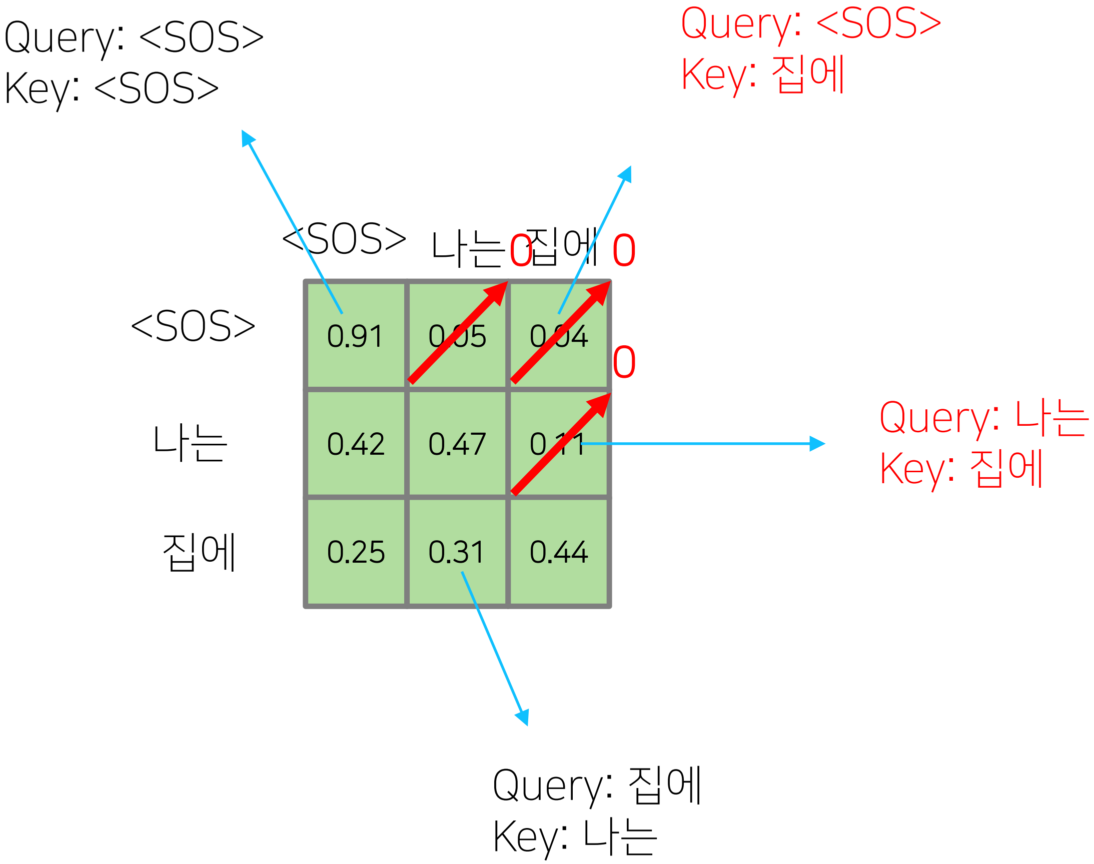
Query: 나는
Key: 집에

Query: 집에
Key: 나는

Transformer: Masked Self-Attention

- Those words not yet generated cannot be accessed during the inference time
- Renormalization of softmax output prevents the model from accessing ungenerated words





Transformer: Experimental Results

- Results on English-German/French translation (newstest2014)

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	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**	**41.8**	$2.3 \cdot 10^{19}$	

Attention Is All You Need, NeurIPS'17

boostcamp AI Tech

References

- Illustrated transformer
    - http://jalammar.github.io/illustrated-transformer/
- The Annotated Transformer
    - http://nlp.seas.harvard.edu/2018/04/03/attention.html
- CS224n –Deep Learning for Natural Language Processing
    - http://web.stanford.edu/class/cs224n/
- Convolution Sequence to Sequence
    - https://arxiv.org/abs/1705.03122
- Fully-parallel text generation for neural machine translation
    - https://blog.einstein.ai/fully-parallel-text-generation-for-neural-machine-translation/

Attention Is All You Need, NeurIPS'17

End of Document
Thank You.