Attention

in Deep Learning: Transformers, visual transformers…

2022

Rufin VanRullen

Directeur de Recherche,

Laboratoire Cerveau et Cognition (CerCo), Unité mixte CNRS-UPS

Attention: definitions

A reminder:

- **Attention is how we select things and discard others**
- **Attention can be bottom-up** (exogenous, world-dependent), **or top-down** (endogenous, task-dependent)

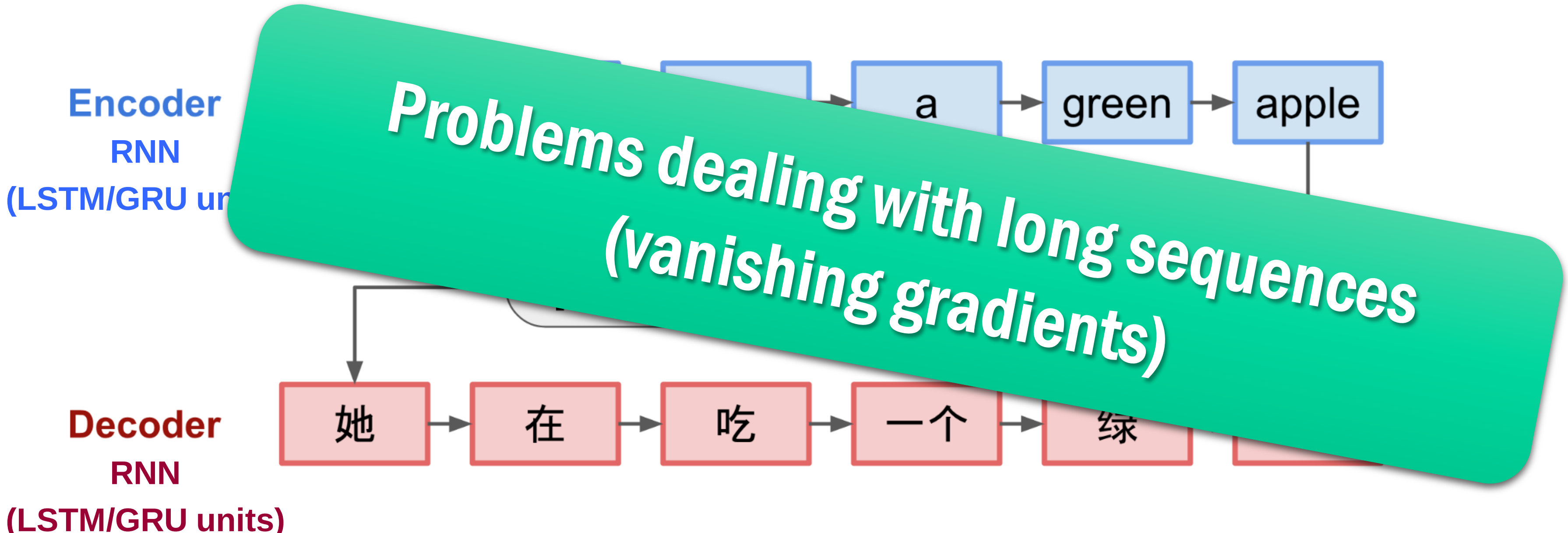
Models of attention

- **Psychological models**
  - Feature-Integration Theory (Treisman)
  - Guided Search (Wolfe)
- **Neuro-Computational models**
  - Saliency map (Itti/Koch)
  - Selective tuning (Tsotsos)
  - Reentry (Hamker)
- **ML (Deep Learning) models**
  - Transformers in NLP (Vaswani)
  - Transformers in computer vision ←code/notebook

Attention in Deep Learning

Sources: Attention? Attention! (blog: lilianweng.github.io/lil-log/2018/06/24/attention-attention.html)
The Illustrated Transformer (blog: jalammar.github.io/illustrated-transformer/)
Self-attention in Computer Vision (blog: towardsdatascience.com/self-attention-in-computer-vision-2782727021f6)

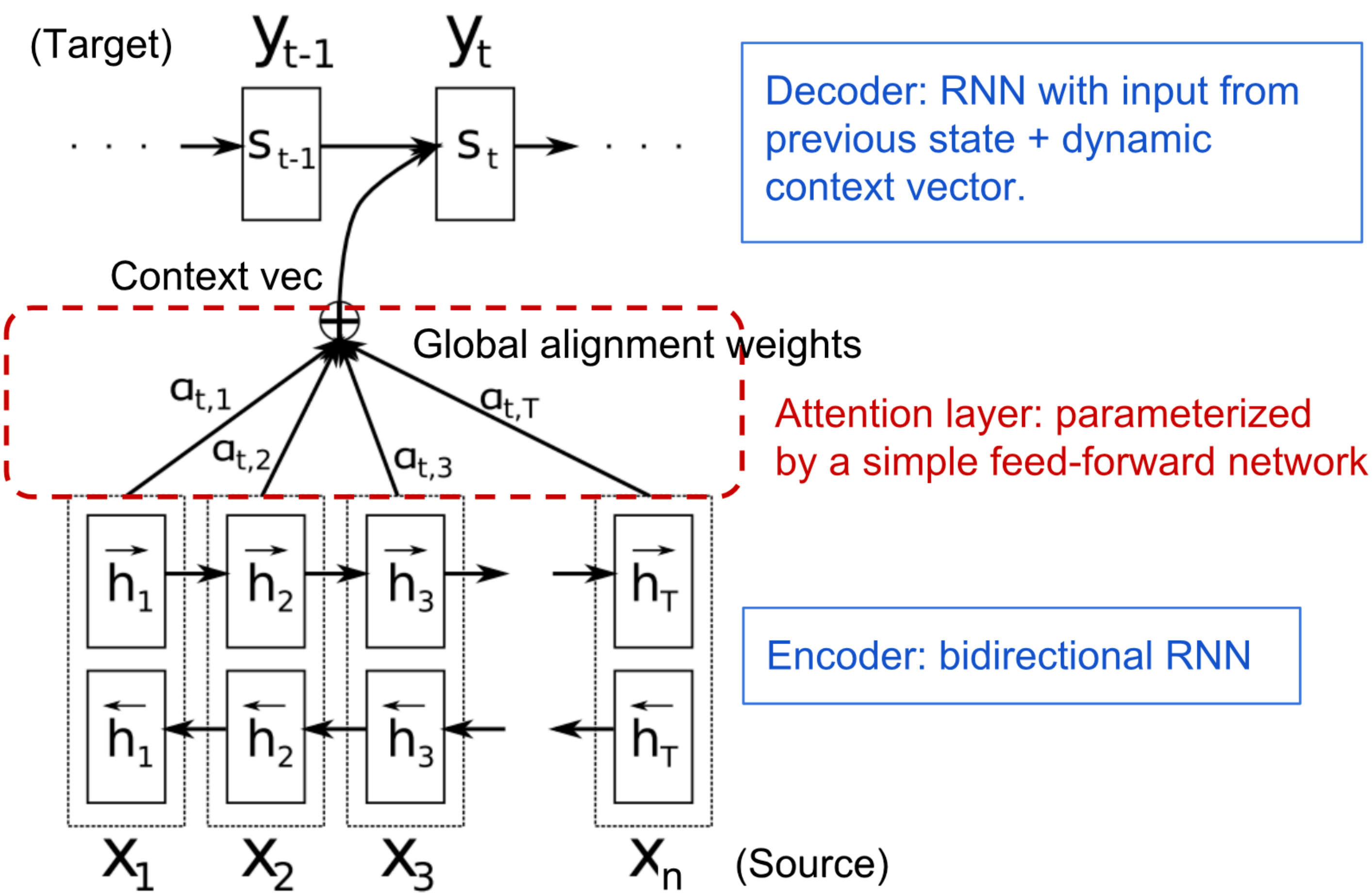
- **Seq2Seq models in Natural Language Processing (NLP)**
  - **Machine translation**
  - **Question-Answer dialog (chatbots)**
  - **Document summarization**
  - **…**



Sutskever et al (2014)

Attention in Deep Learning

- **Solving the vanishing gradients problem with attention**



Bahdanau et al (2015)

Attention in Deep Learning

- **Solving the vanishing gradients problem with attention**

Source sequence $\mathbf{x} = [x_1, x_2, \dots, x_n]$

Output Sequence $\mathbf{y} = [y_1, y_2, \dots, y_m]$

Attention score: $\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$

where both $\mathbf{v}_a$ and $\mathbf{W}_a$ are weight matrices to be learned

Input: sequence

Output: sequence

Attention: =a form of memory pointer?

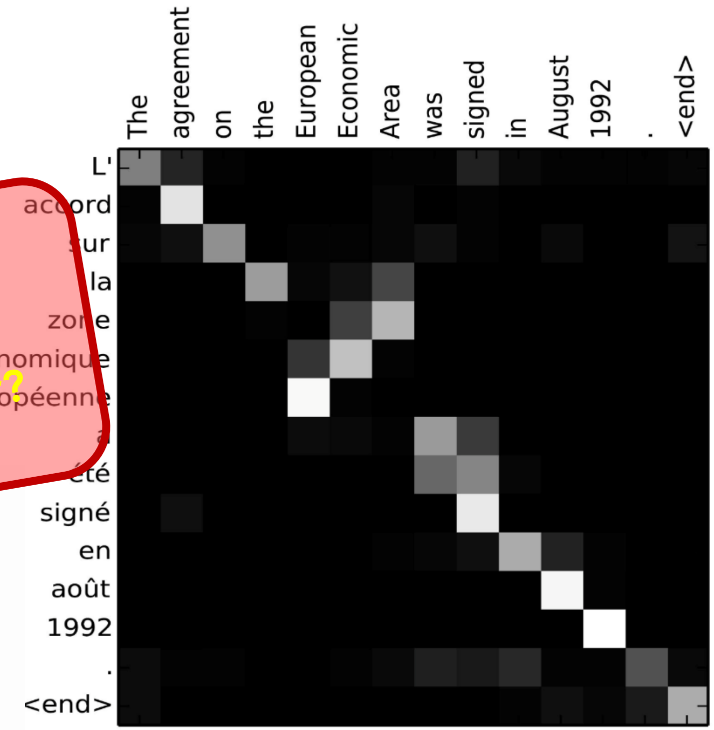
The decoder network has hidden state $\boldsymbol{s}_t = f(\boldsymbol{s}_{t-1}, y_{t-1}, \mathbf{c}_t)$ for the output word at position t, $t = 1, \dots, m$, where the context vector $\mathbf{c}_t$ is a sum of hidden states of the input sequence, weighted by alignment scores:

$$\mathbf{c}_t = \sum_{i=1}^{n} \alpha_{t,i} \boldsymbol{h}_i \quad \text{; Context vector for output } y_t$$

$$\alpha_{t,i} = \text{align}(y_t, x_i) \quad \text{; How well two words } y_t \text{ and } x_i \text{ are aligned.}$$

$$= \frac{\exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))}{\sum_{i'=1}^{n} \exp(\text{score}(\boldsymbol{s}_{t-1}, \boldsymbol{h}_{i'}))} \quad \text{; Softmax of some predefined alignment score..}$$

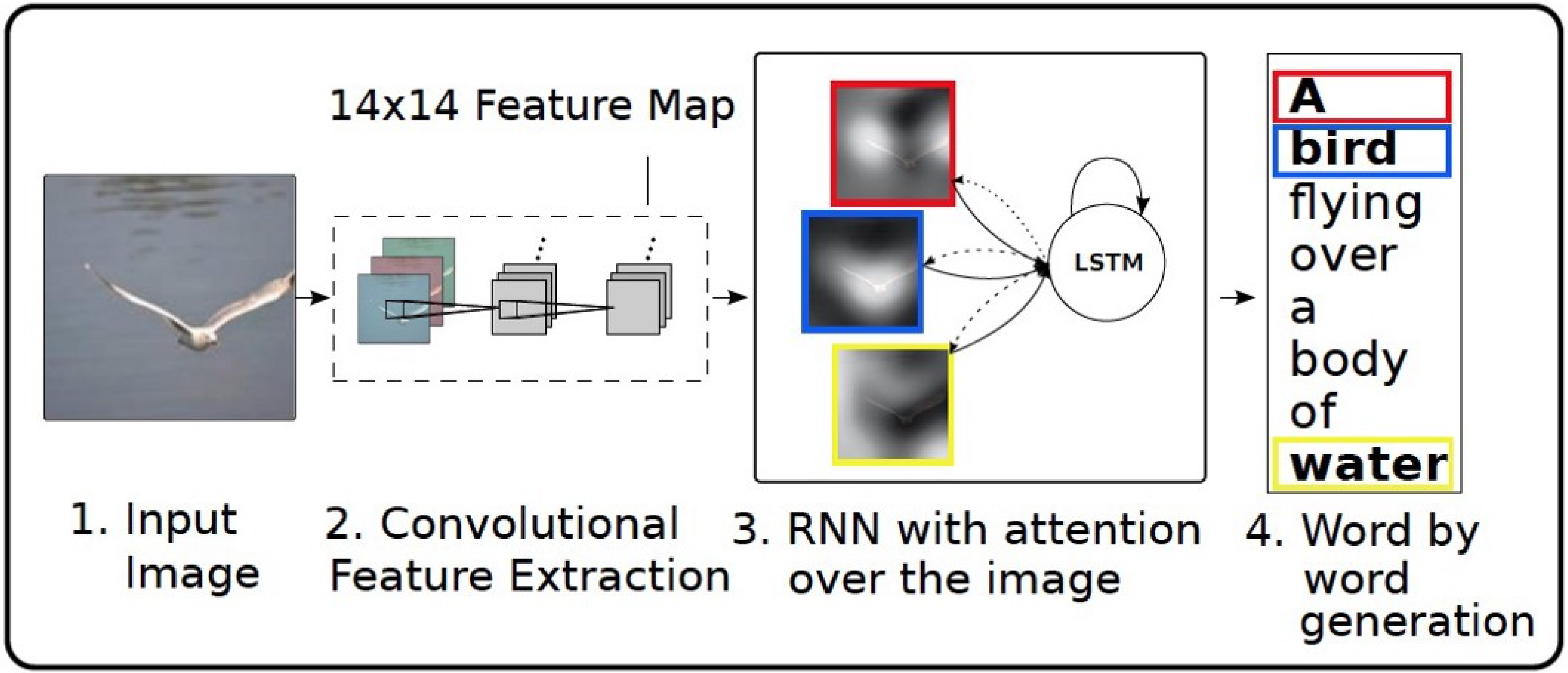
French → English translation



Bahdanau et al (2015)

Attention in Deep Learning

- **Show, attend and tell**



Xu et al (2015)

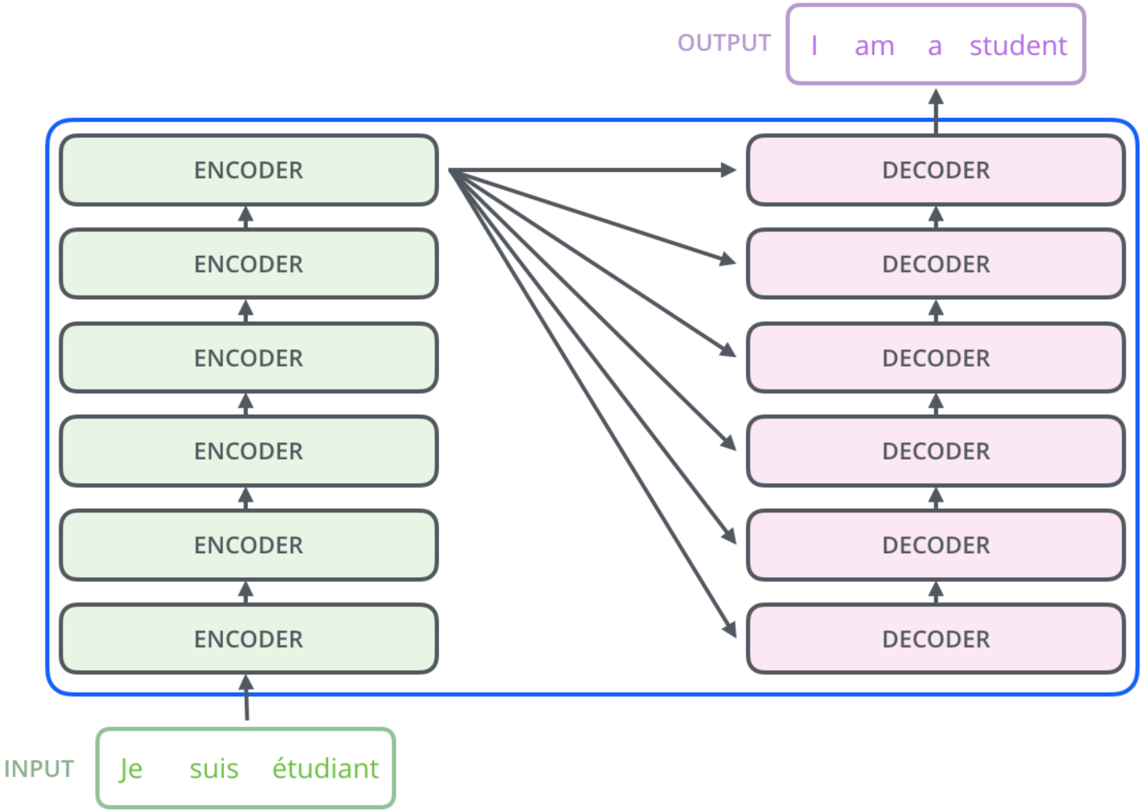
Attention in Deep Learning

- **Show, attend and tell**

Xu et al (2015)

Attention in Deep Learning

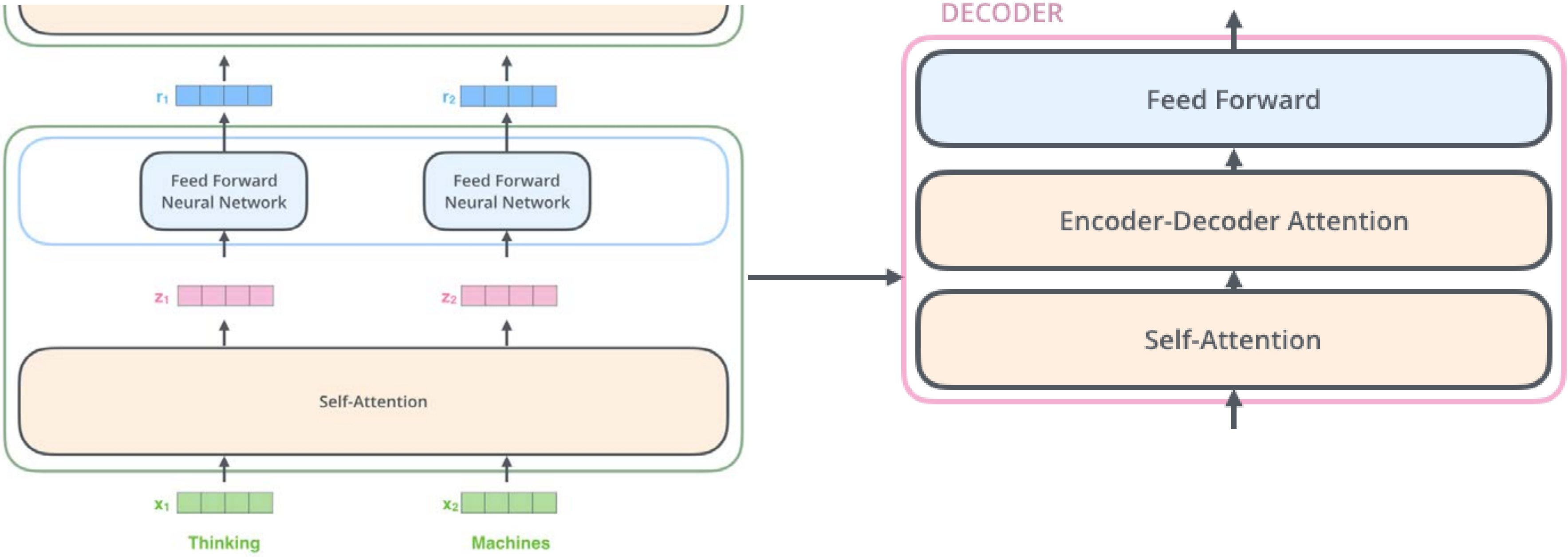
- **Back to NLP: Attention is all you need (Transformers)!**
  - Main idea: no recurrence, feed-forward architecture, all inputs are provided at once (very large matrix)
  - Attention takes care of long-range dependencies



Vaswani et al (2017)

Attention in Deep Learning

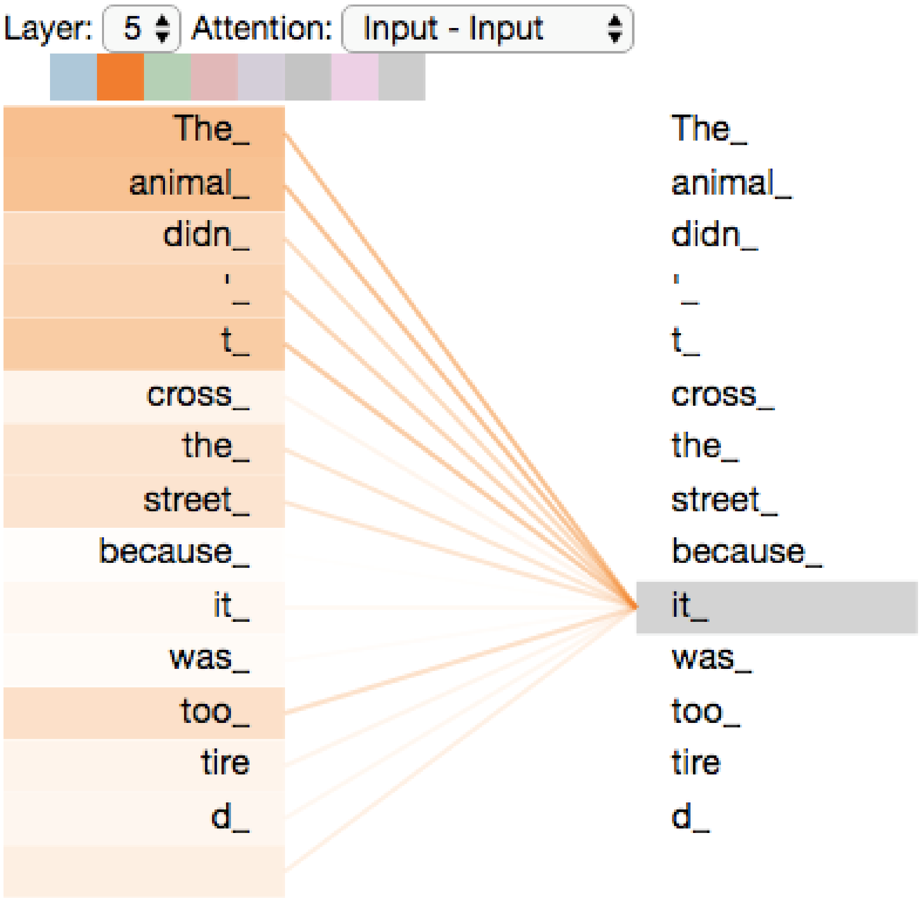
- **Transformer details**
  - Self-attention layer: every token/position can « attend » to every other in the <u>same</u> layer
  - Encoder-decoder-attention: every position in the decoder can attend to every position in the final <u>encoder</u> layer (as in Seq2Seq models)
  - The same feed-forward network is applied independently to each position



Vaswani et al (2017)

Attention in Deep Learning

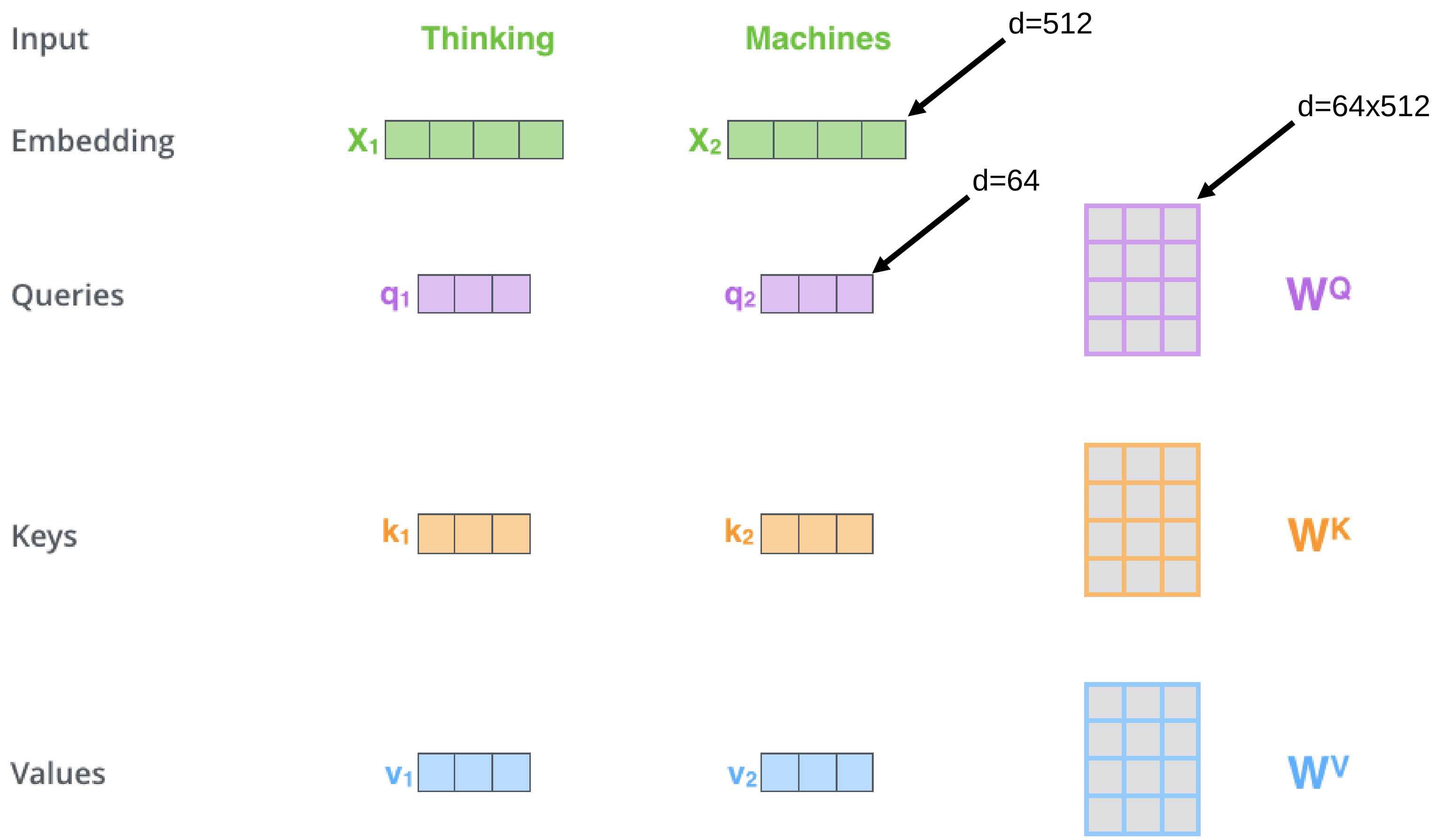
- **Transformer details**
  - Self-attention example



Vaswani et al (2017)

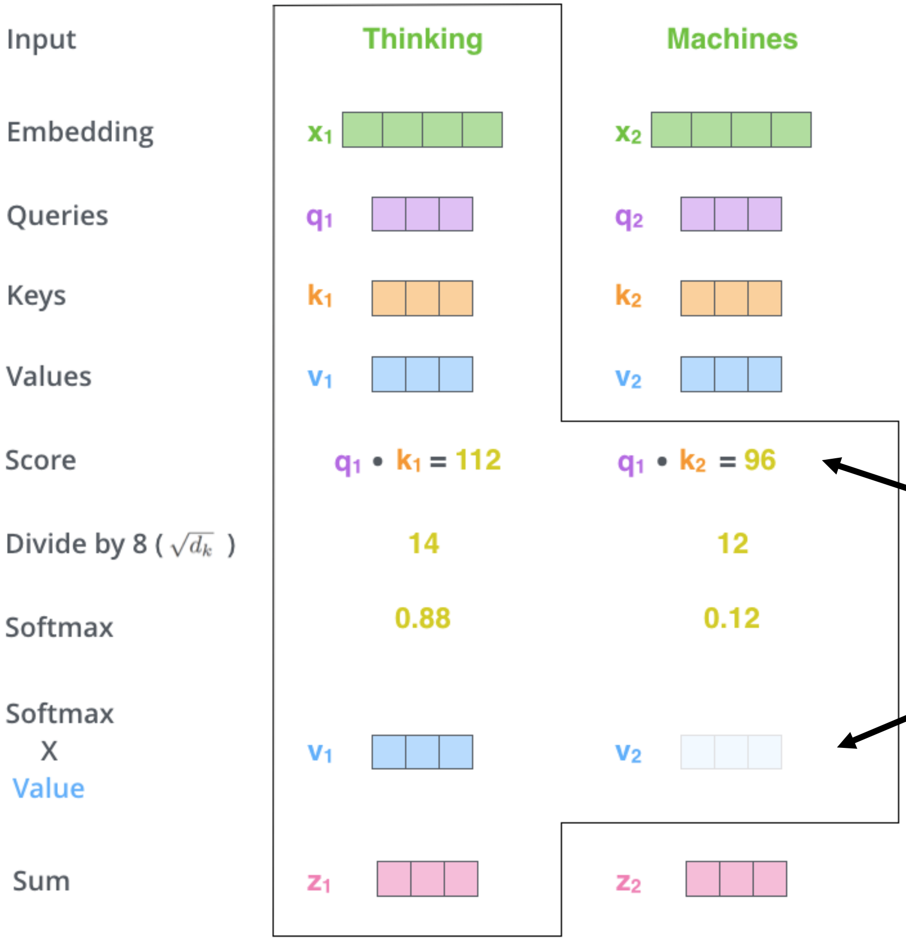
Attention in Deep Learning

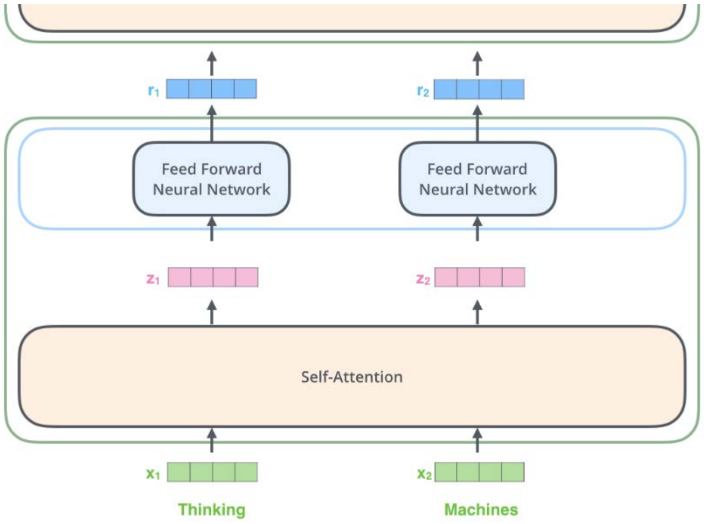
- **Transformer details**
  - Self-attention computation: Queries, Keys, Values



Vaswani et al (2017)

- **Transformer details**
  - Self-attention computation: Queries, Keys, Values



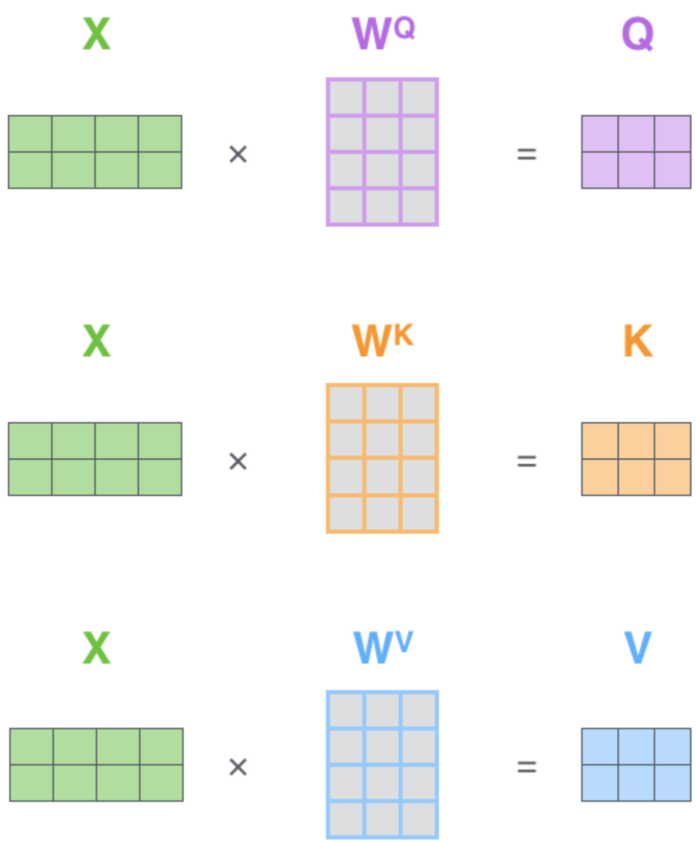


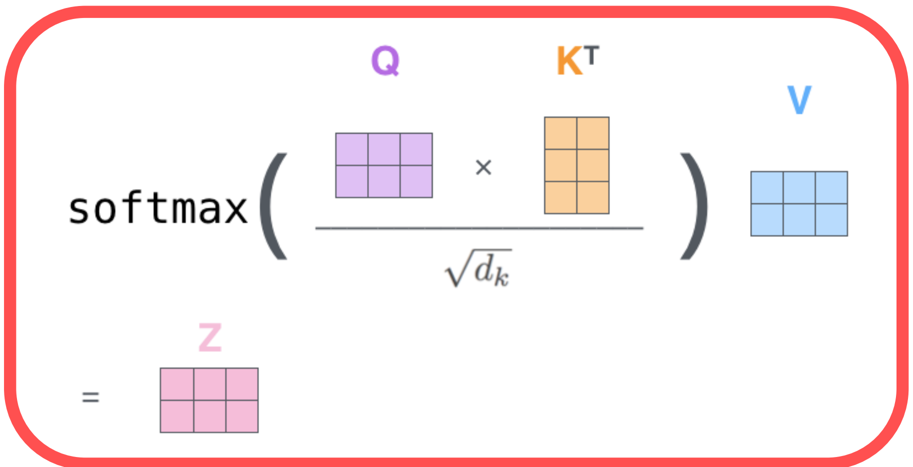
The match between ***query*** and ***key*** determines how much of each ***value*** is included in the final output

Vaswani et al (2017)

Attention in Deep Learning

- **Transformer details**
  - Self-attention computation: matrix version



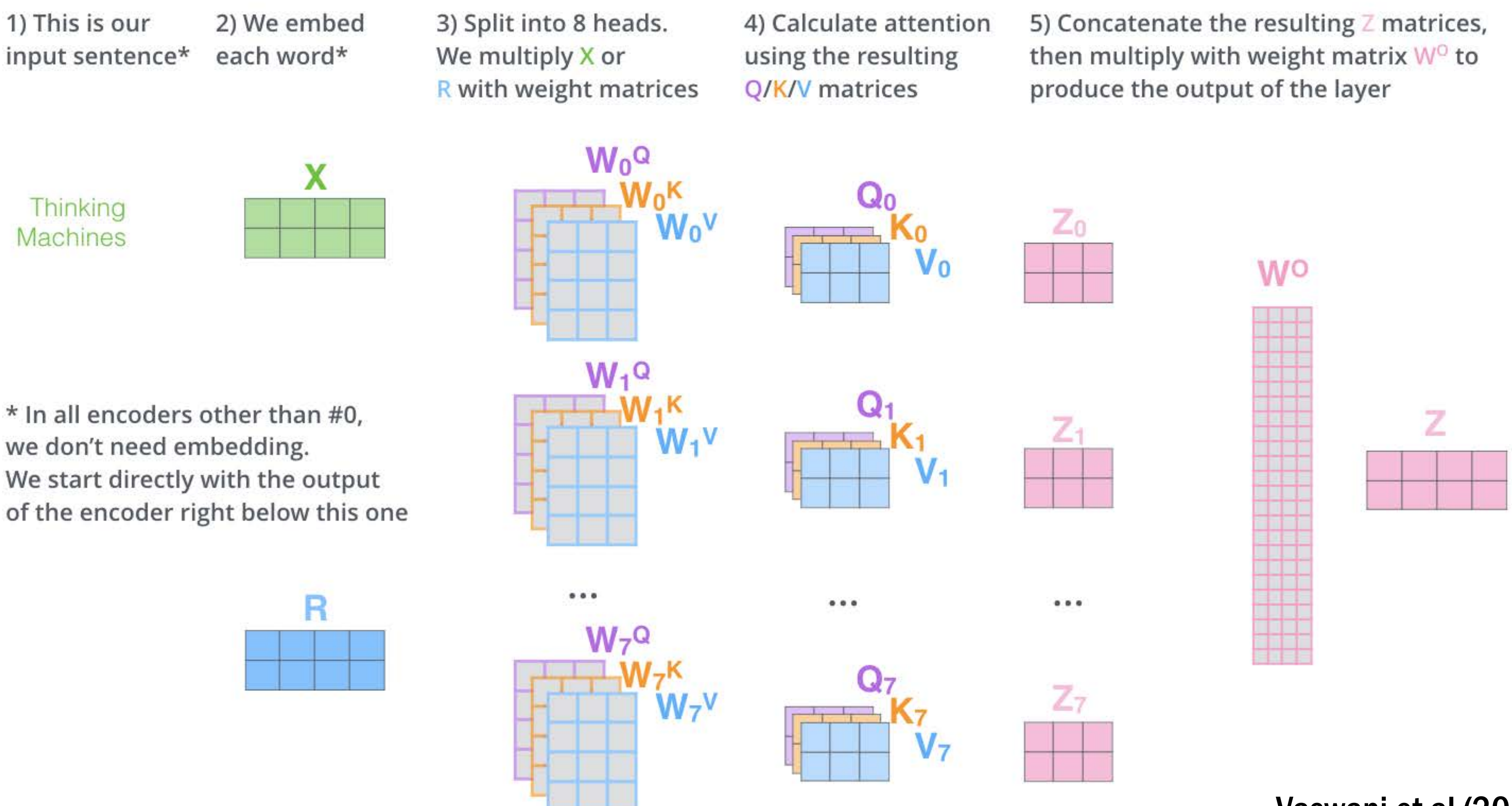


Pay *attention*… This is what we'll implement in the notebook!

Vaswani et al (2017)

Attention in Deep Learning

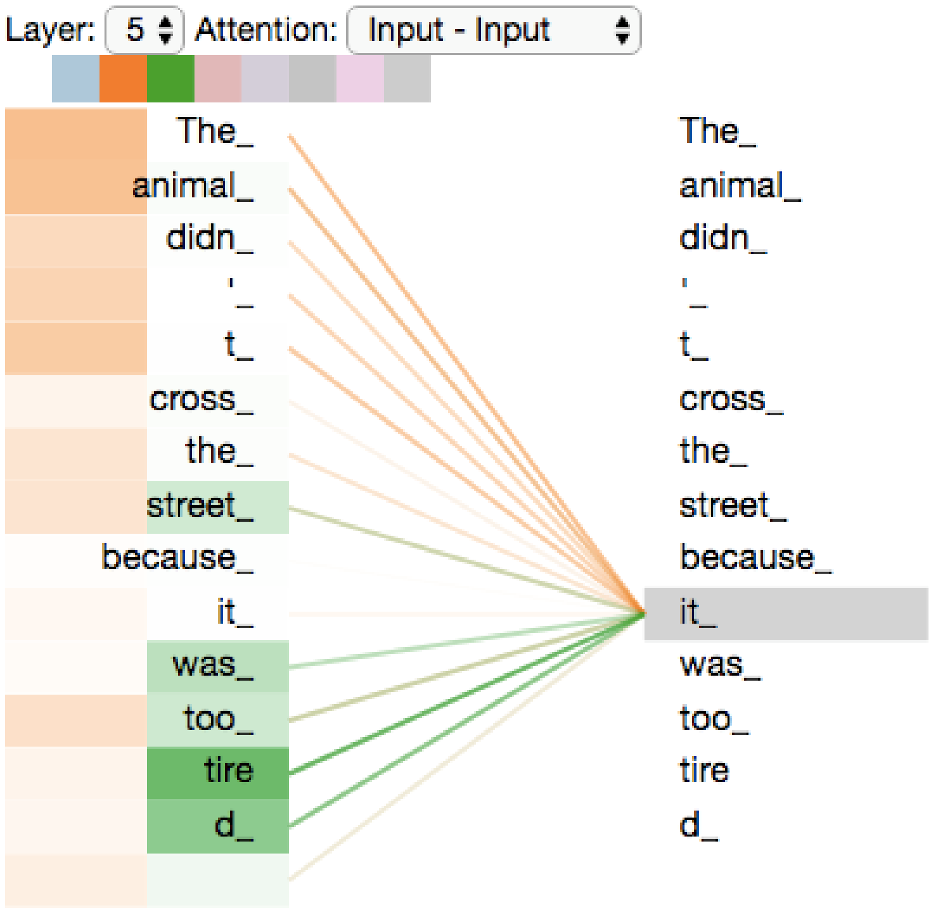
- **Transformer details**
  - Multi-head attention



Vaswani et al (2017)

Attention in Deep Learning

- **Transformer details**
  - Multi-head attention

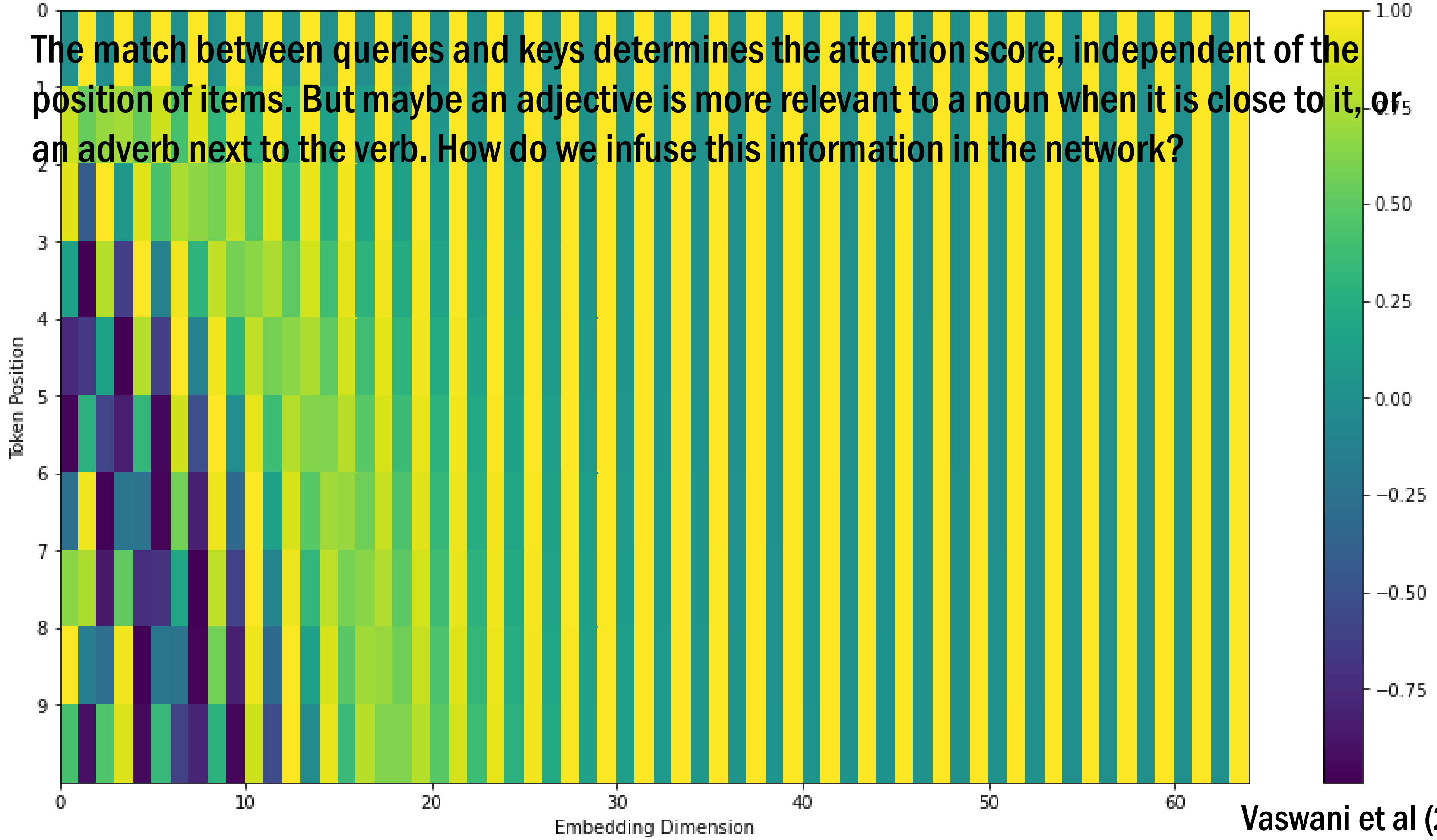


Vaswani et al (2017)

Attention in Deep Learning

- **Transformer details**
  - Positional encoding

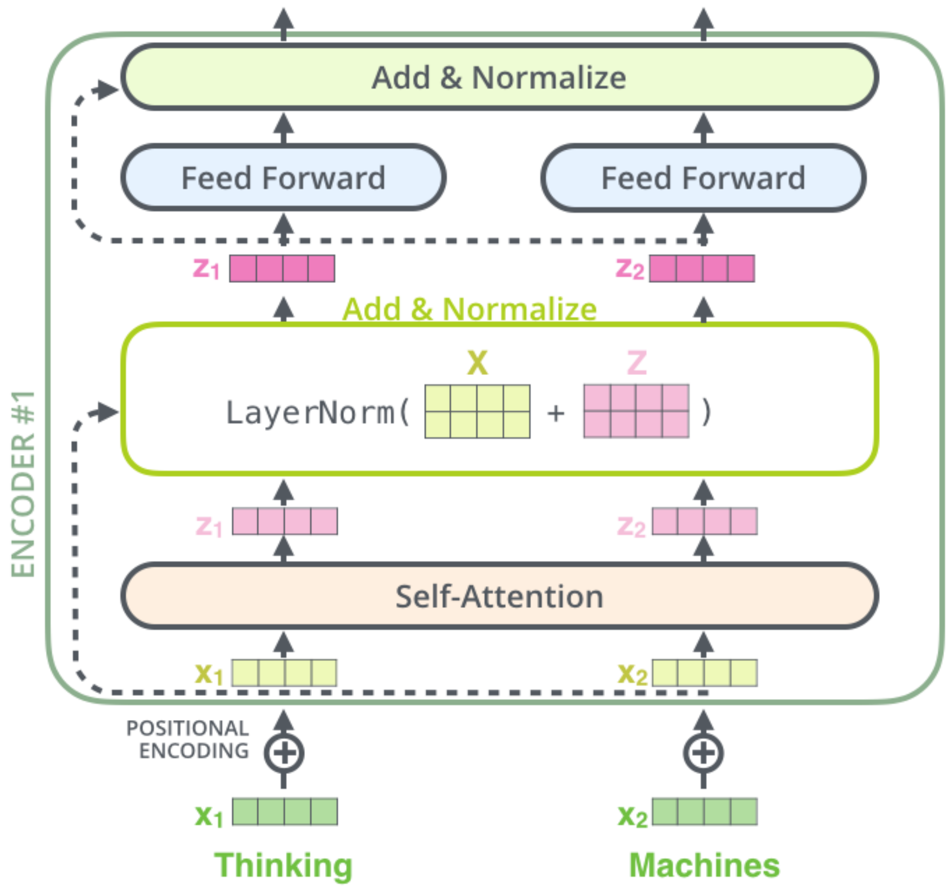
The match between queries and keys determines the attention score, independent of the position of items. But maybe an adjective is more relevant to a noun when it is close to it, or an adverb next to the verb. How do we infuse this information in the network?



Vaswani et al (2017)

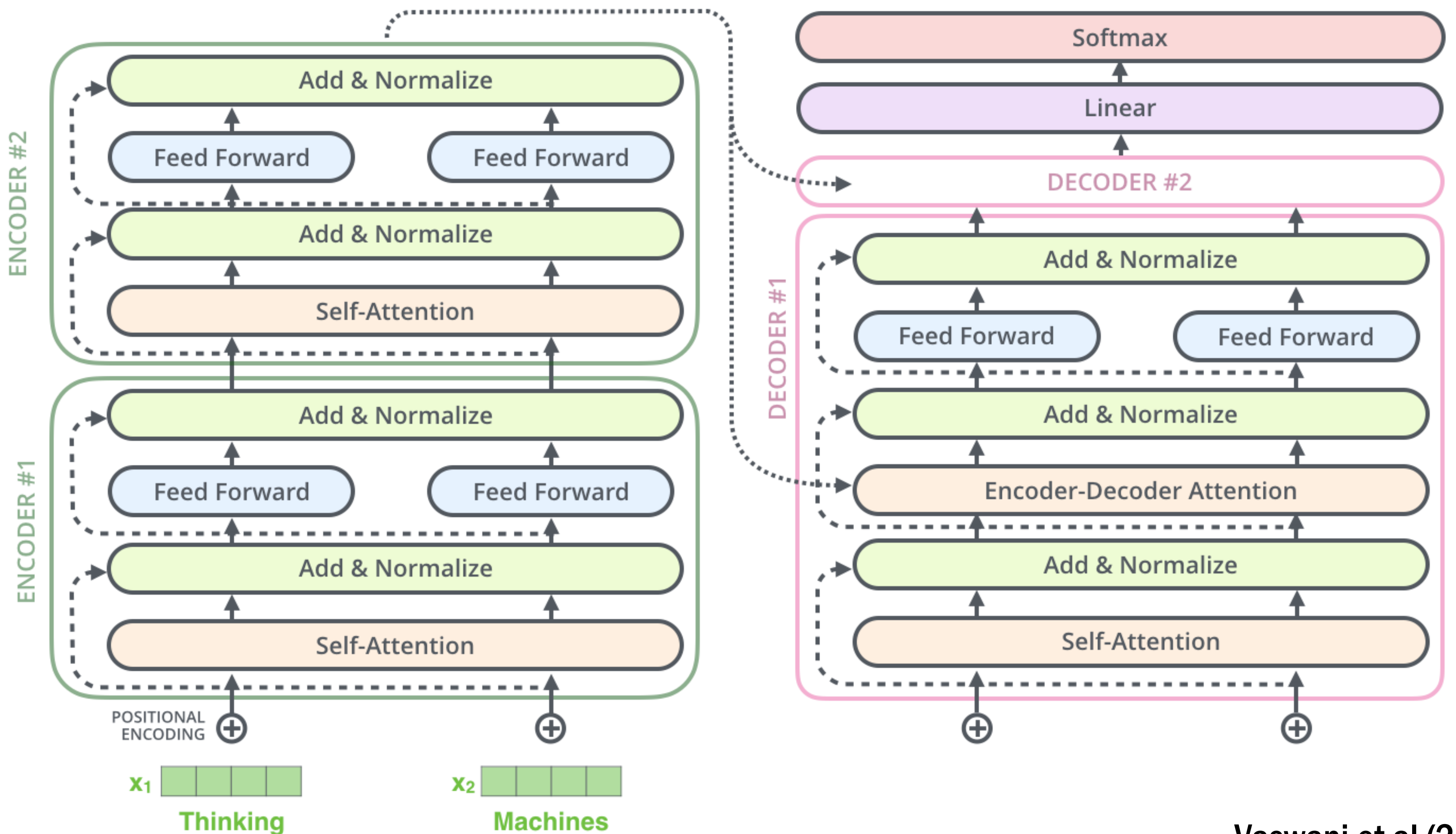
Attention in Deep Learning

- **Transformer details**
  - Residual connections



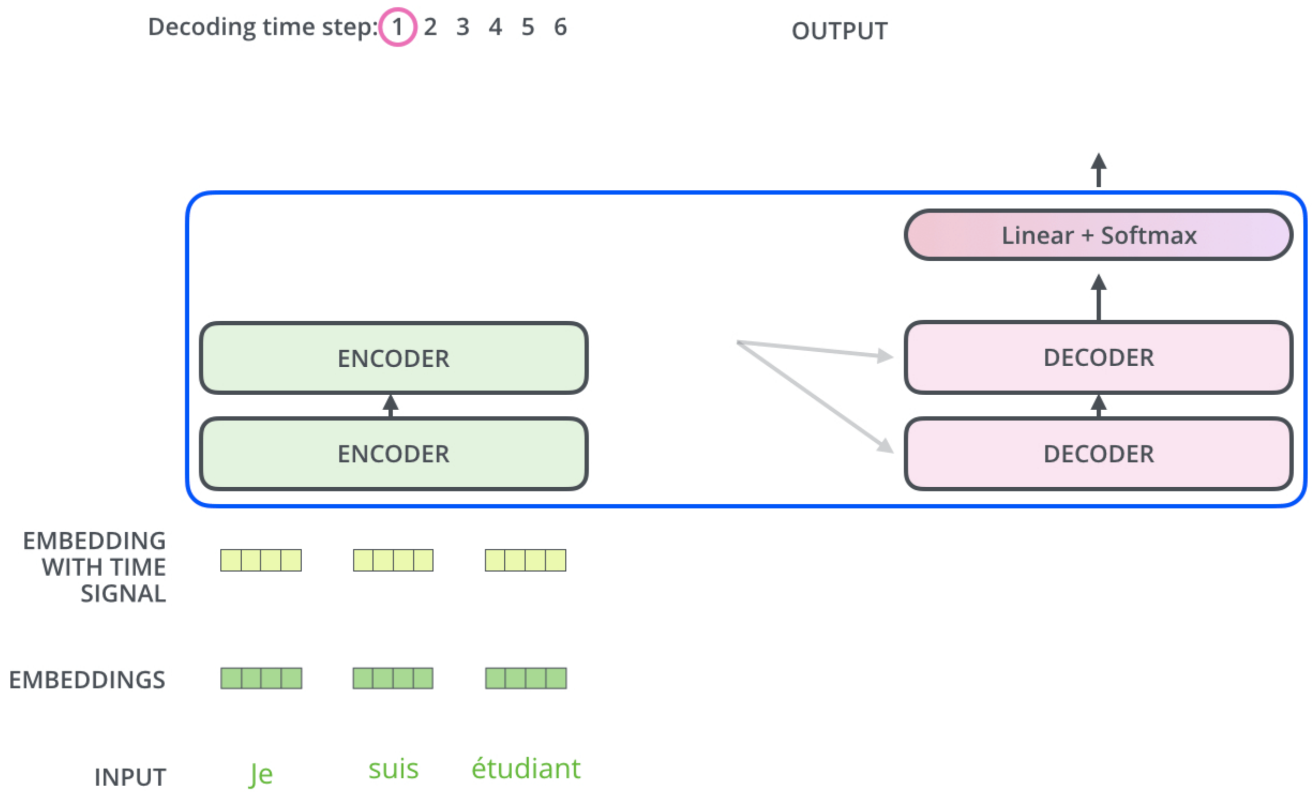
Vaswani et al (2017)

- **Transformer details**
  - Encoder + Decoder



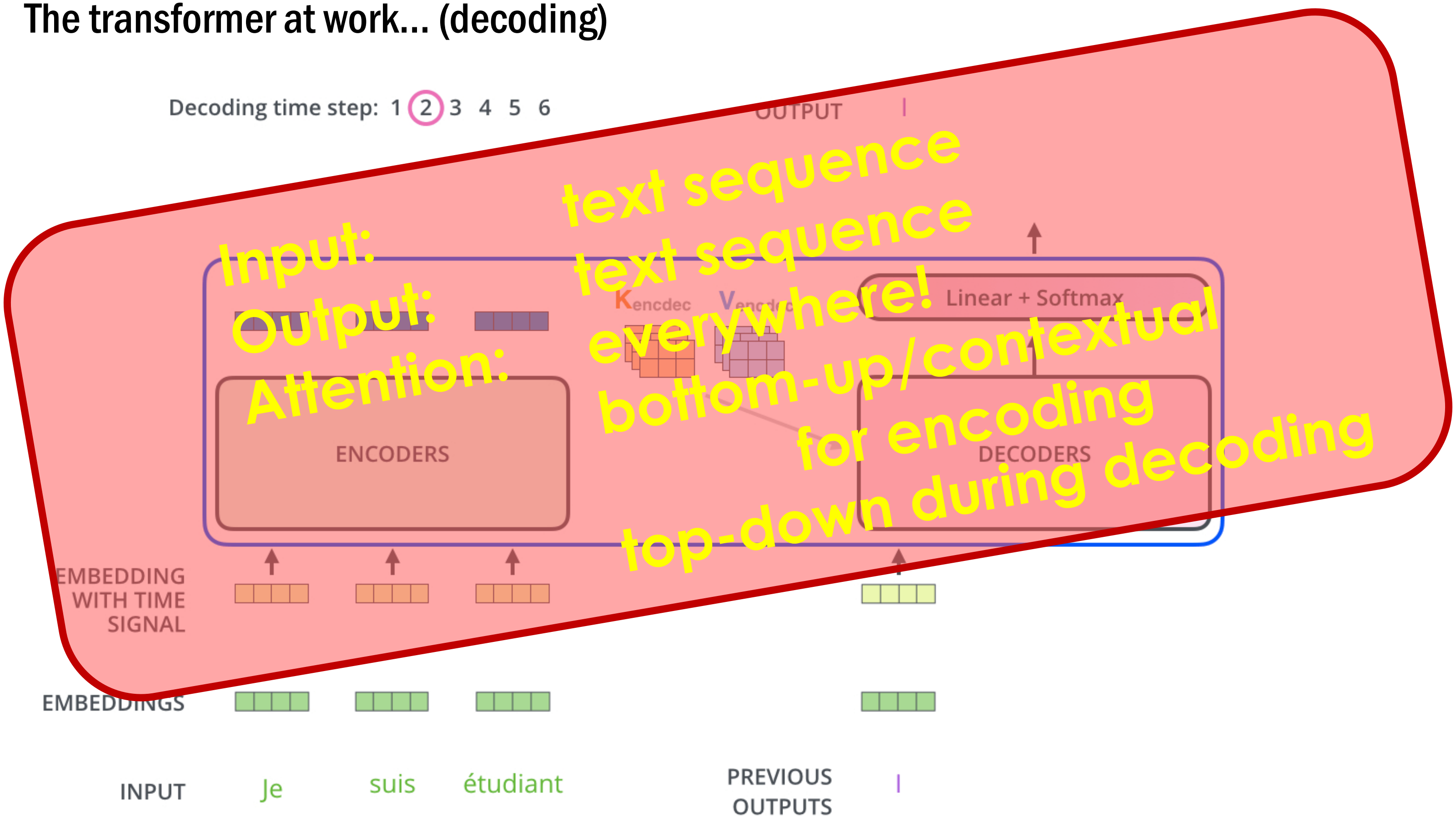
Vaswani et al (2017)

- **Transformer details**
  - The transformer at work… (encoding)



Vaswani et al (2017)

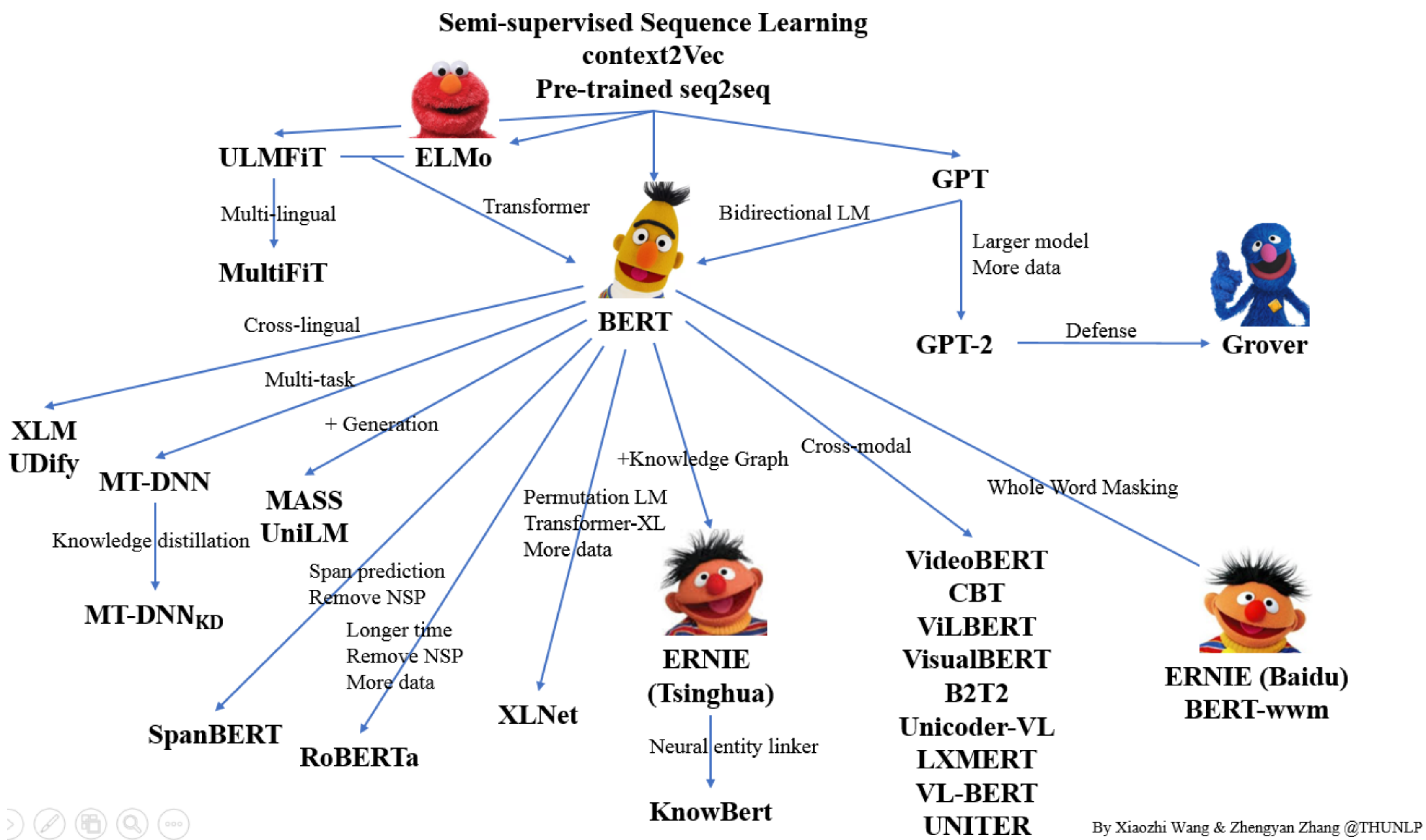
- **Transformer details**
  - The transformer at work… (decoding)



Vaswani et al (2017)

Attention in Deep Learning

- **Family tree of NLP Transformers:**



Attention in Deep Learning

- **Transformers (attention) in computer vision**
  - Many recent papers have thought to augment or replace convolution operations in deep networks with a self-attention mechanism
  - One example: Ramachandran et al (2019), Stand-alone self-attention in vision models, NeurIPS.

Single-headed attention for computing the pixel output $y_{ij} \in \mathbb{R}^{d_{out}}$ is then computed as follows (see Figure 3):

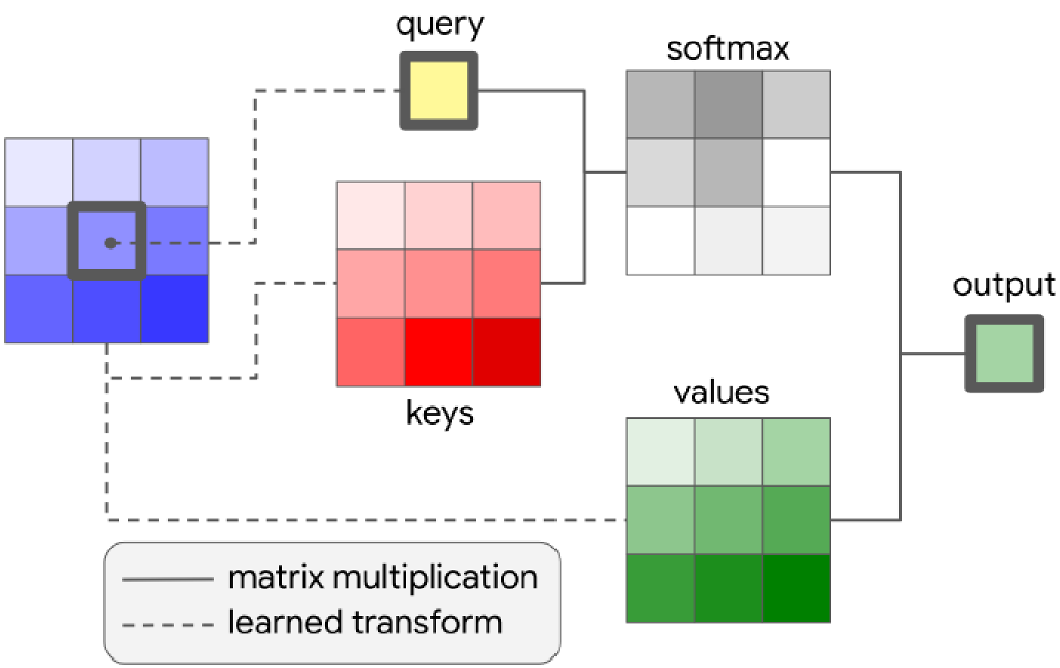
$$y_{ij} = \sum_{a,b \in \mathcal{N}_k(i,j)} \texttt{softmax}_{ab}\left(q_{ij}^\top k_{ab}\right) v_{ab} \qquad (2)$$

where the *queries* $q_{ij} = W_Q x_{ij}$, *keys* $k_{ab} = W_K x_{ab}$, and *values* $v_{ab} = W_V x_{ab}$ are linear transformations of the pixel in position ij and the neighborhood pixels. $\texttt{softmax}_{ab}$ denotes a softmax applied to all logits computed in the neighborhood of ij. $W_Q, W_K, W_V \in \mathbb{R}^{d_{out} \times d_{in}}$ are all learned transforms.

Pay *attention*… This is what we'll implement in the notebook!

Attention in Deep Learning

- **Transformers (attention) in computer vision**
  - Many recent papers have thought to augment or replace convolution operations in deep networks with a self-attention mechanism
  - One example: Ramachandran et al (2019), Stand-alone self-attention in vision models, NeurIPS.



Attention in Deep Learning

- **Transformers (attention) in computer vision**
  - Many recent papers have thought to augment or replace convolution operations in deep networks with a self-attention mechanism
  - One example: Ramachandran et al (2019), Stand-alone self-attention in vision models, NeurIPS.
  - Attention is permutation-invariant ➔ position encoding (as in NLP transformers)

$$y_{ij} = \sum_{a,b \in \mathcal{N}_k(i,j)} \texttt{softmax}_{ab} \left(q_{ij}^\top k_{ab} + q_{ij}^\top r_{a-i,b-j} \right) v_{ab} \tag{3}$$

Thus, the logit measuring the similarity between the query and an element in $\mathcal{N}_k(i, j)$ is modulated both by the content of the element and the relative distance of the element from the query. Note that by infusing relative position information, self-attention also enjoys translation equivariance, similar to convolutions.

Attention in Deep Learning

- **Transformers (attention) in computer vision**
  - Many recent papers have thought to augment or replace convolution operations in deep networks with a self-attention mechanism
  - One example: Ramachandran et al (2019), Stand-alone self-attention in vision models, NeurIPS.
  - Accuracy on ImageNet (ResNet50 backbone, varying channel numbers):
  - Comparable performance with ½ the parameters!

Input: image
Output: class label
Attention: (everywhere!) bottom-up/contextual

Attention in Deep Learning

- **Transformers (attention) in computer vision**

- Colab Notebook:

https://github.com/rufinv/DL-Attention-Class/blob/main/VanRullen_AttentionClass_CIFAR10_2022.ipynb

Implement, train and test a
Deep Convolutional Neural Network (CNN)
for image classification (CIFAR)
with/without attention

go to link now…