



07 – TRANSFORMER ARCHITECTURE

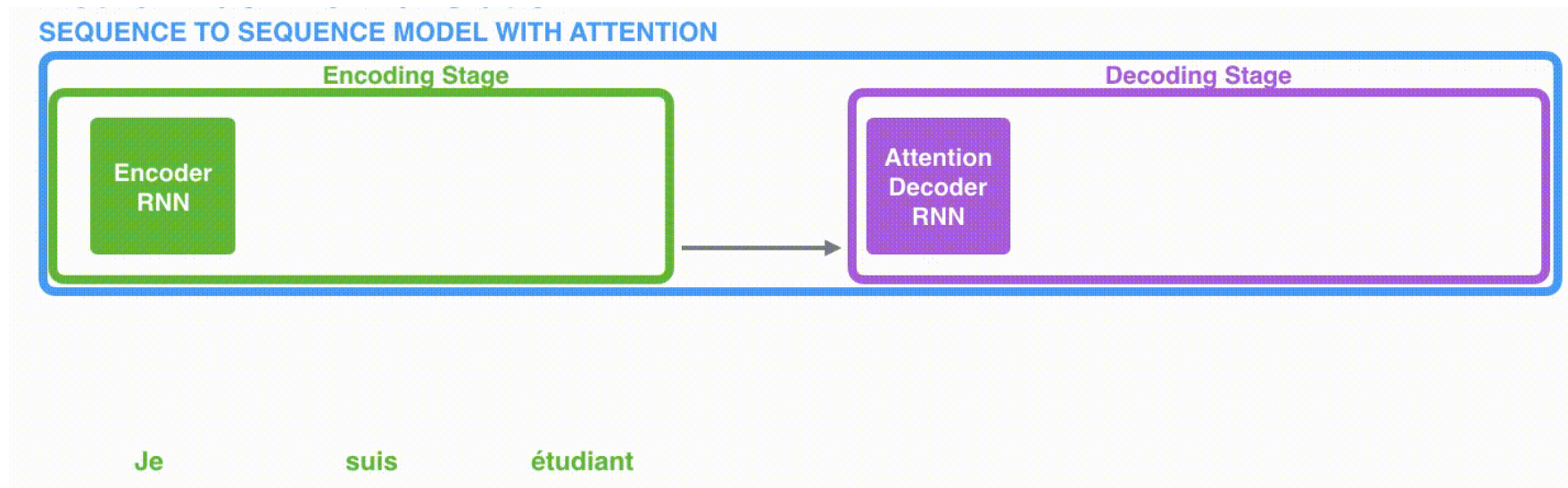
MACHINE LEARNING FOR NATURAL LANGUAGE PROCESSING, AIMS 2024

Lecture 07
Dr. Elvis Ndah

OVERVIEW

1. What is a Transformer
2. Transformer architecture
3. How transformers work
4. How to calculate self-attention
5. Multi-head attention
6. Transfer Learning

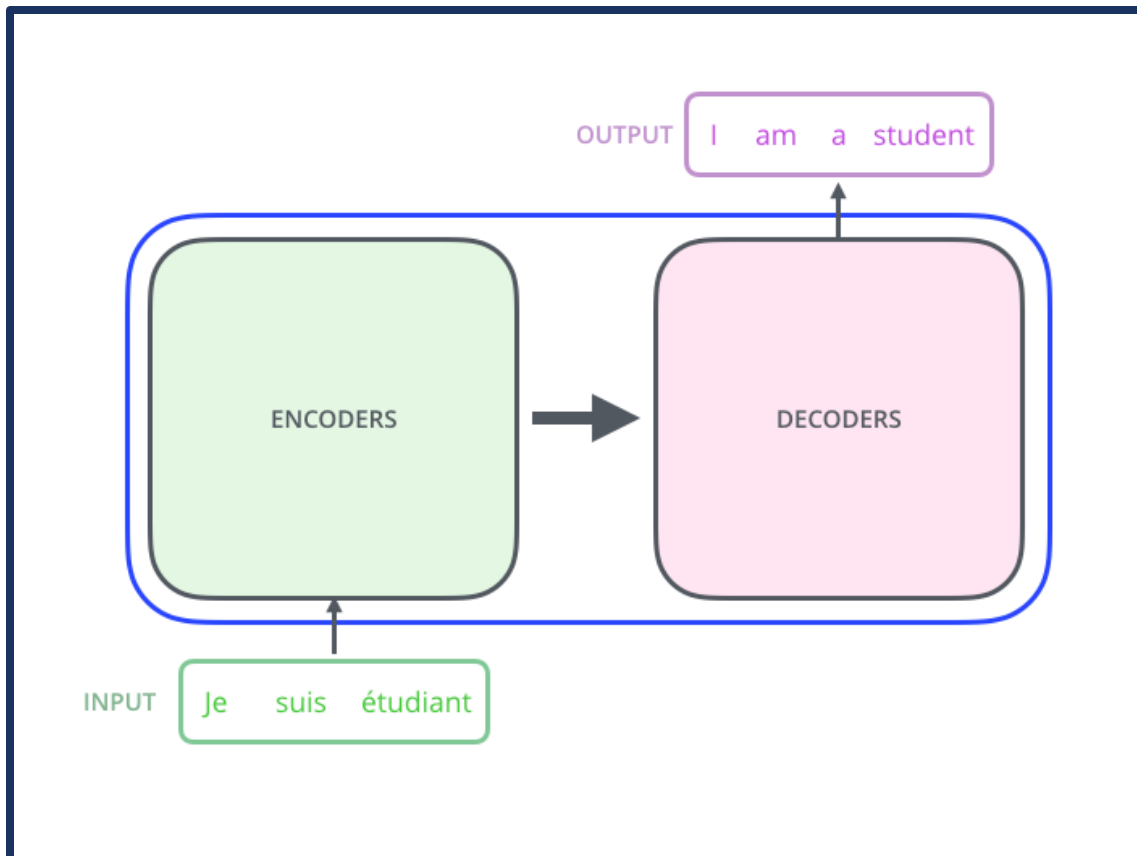
WHAT IS A TRANSFORMER



WHAT IS A TRANSFORMER



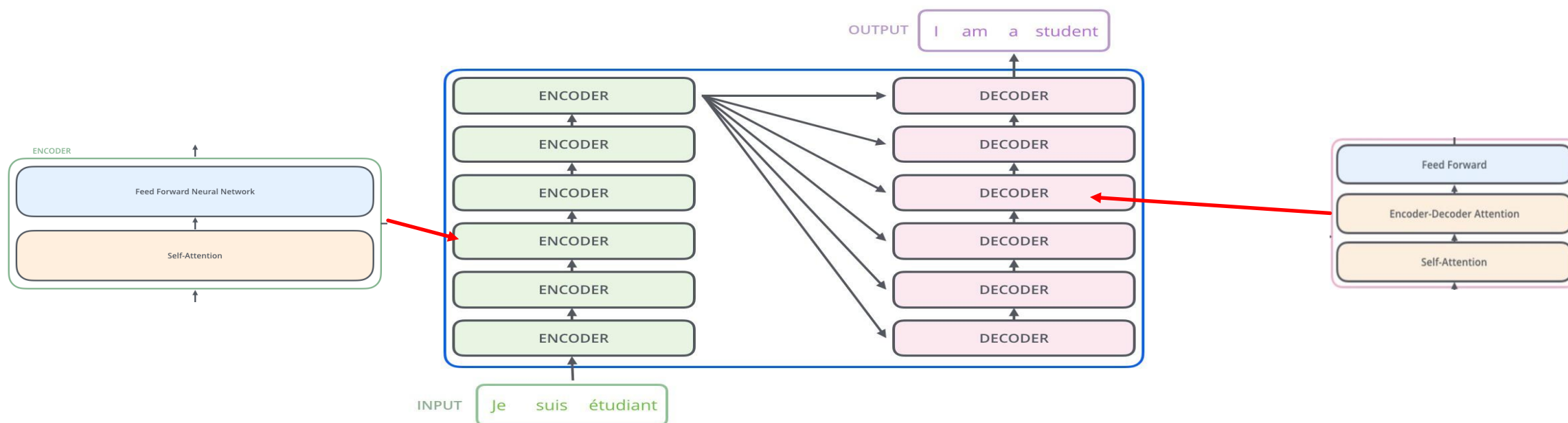
TRANSFORMERS



- **Transformer** – proposed by Vaswani A., 2017 (Attention is all you need)
 - uses attention to boost the speed with which these models can be trained.
 - The transformer is based on a feedforward neural network rather than RNN.
 - The biggest benefit of the Transformer is the speed of computation due to parallelization.
 - All input sequences are processed simultaneously.

TRANSFORMER – ARCHITECTURE

- Architecture: stacked encoder – decoder architecture
- Encoding component: stack of **encoders** (original paper had 6 encoders)
- **encoders are all identical in structure**
- Decoding component: stack of **decoders** (same number as the encoder)



TRANSFORMER – ARCHITECTURE



Encoder

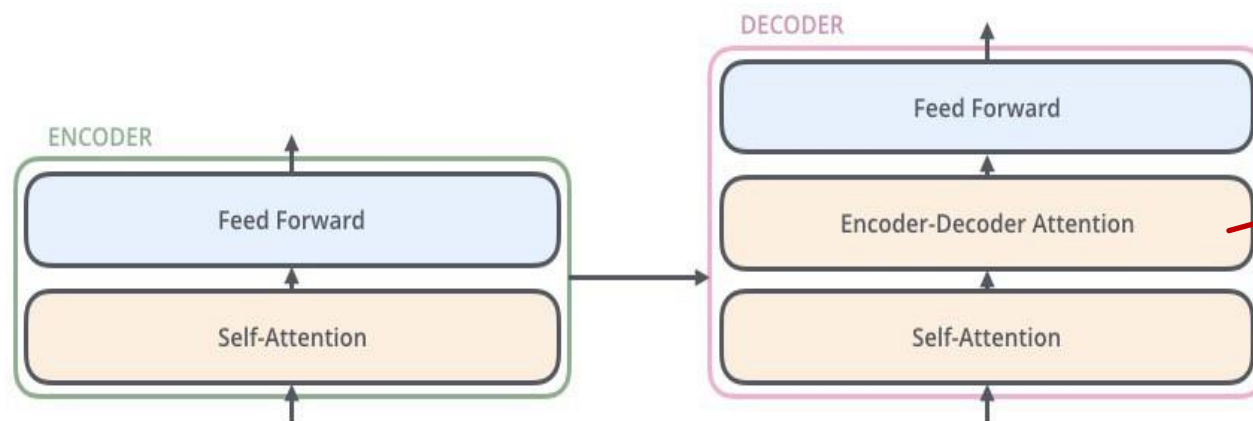
Inputs first flow through a self-attention layer

Outputs of the self-attention layer are fed to a feed-forward neural network.



Decoder

Input from encoder flows through self-attention layer then Encoder-Decoder Attention layer and finally through Feedforward layer

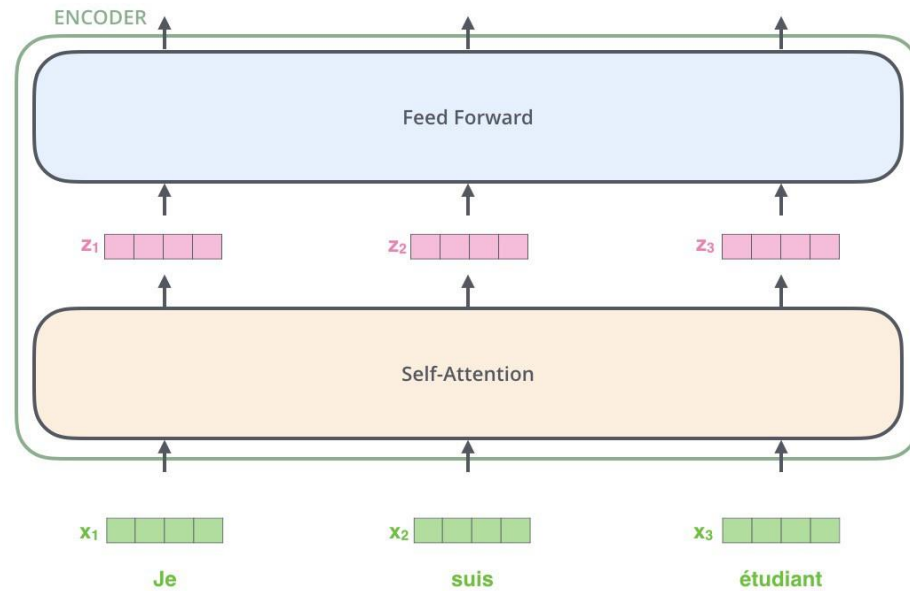


Attention layer:

- helps decoder focus on relevant parts of the input

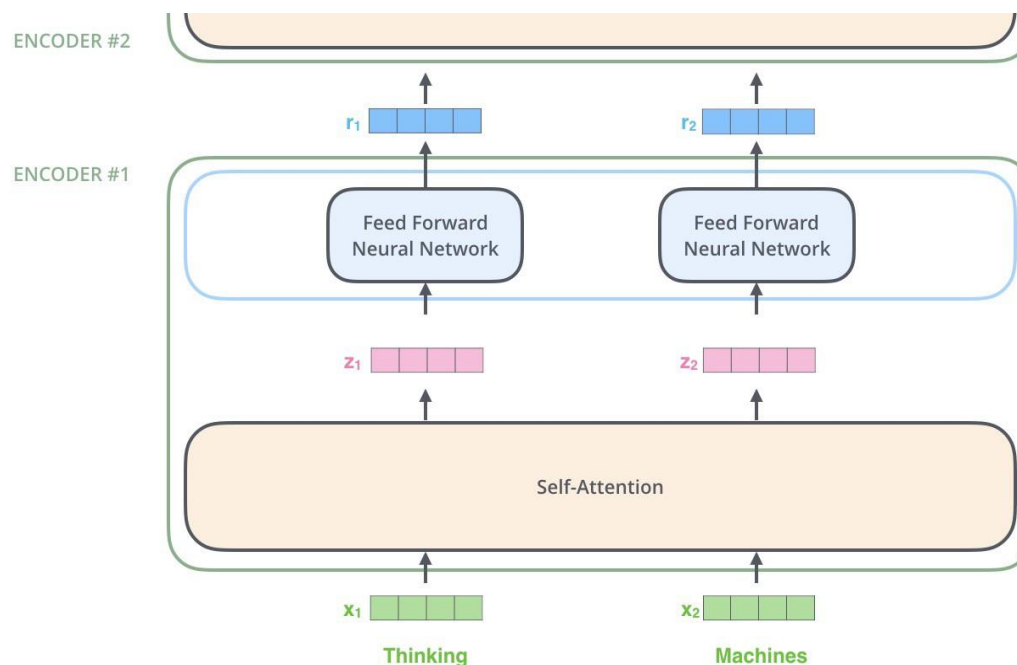
HOW THE TRANSFORMER WORKS

- Convert input word into a word embedding vectors
- Each word in the sentence flows through its own path in the encoder



HOW THE TRANSFORMER WORKS – ENCODING

- Encoder receive input words as vectors
- Self attention processes the input and sent to the feedforward network
- Feedforward sends output to next encoder layer.

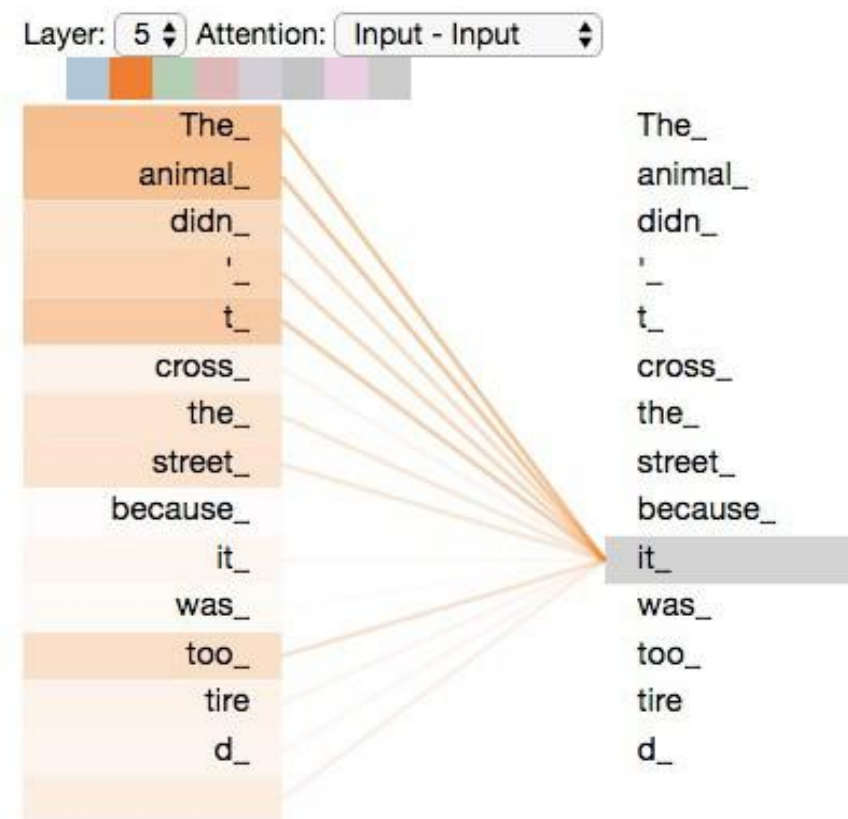


TRANSFORMER – WHAT IS SELF ATTENTION?

Example:

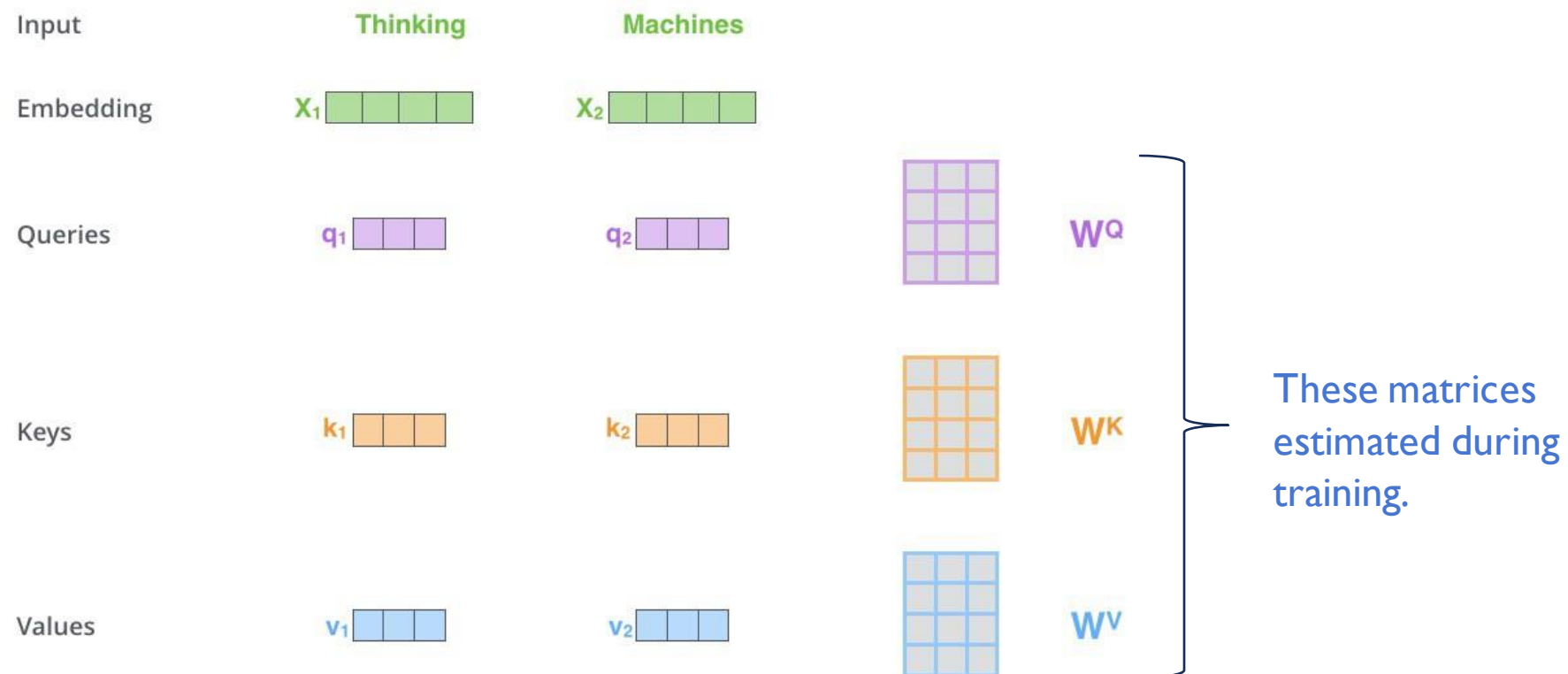
The animal didn't cross the street because it was too tired.

- What does *it* in this sentence refer to?
 - the street*
 - the animal*
- self-attention allows the transformer model to associate
 - it* with *animal*
- self attention allows it to look at other positions for clues to help the model better encode the word *it*



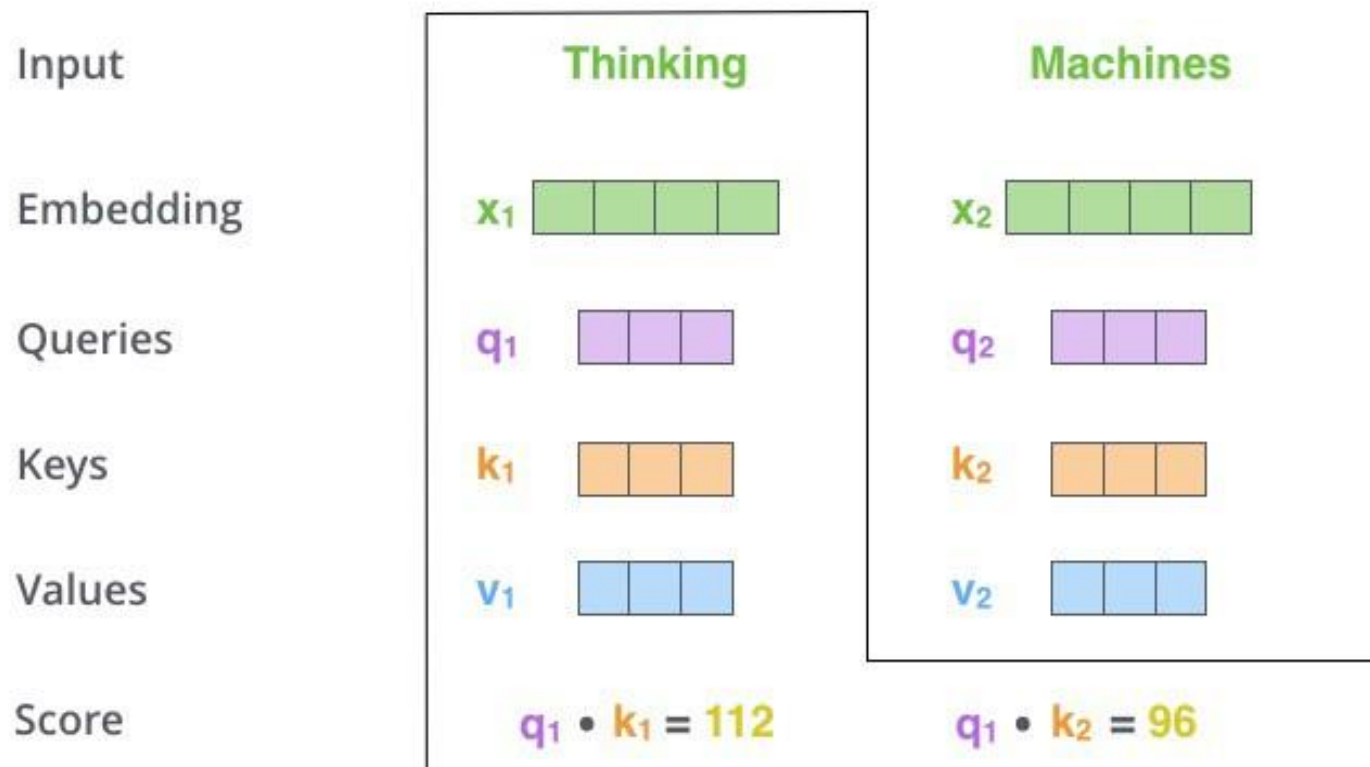
HOW TO CALCULATE SELF ATTENTION

- Step 1: Create 3 vectors (**Q**uery, **k**ey, **V**alue – dimension 64) from each encoder's input vector



HOW TO CALCULATE SELF ATTENTION

- Step 2: calculate self-attention score



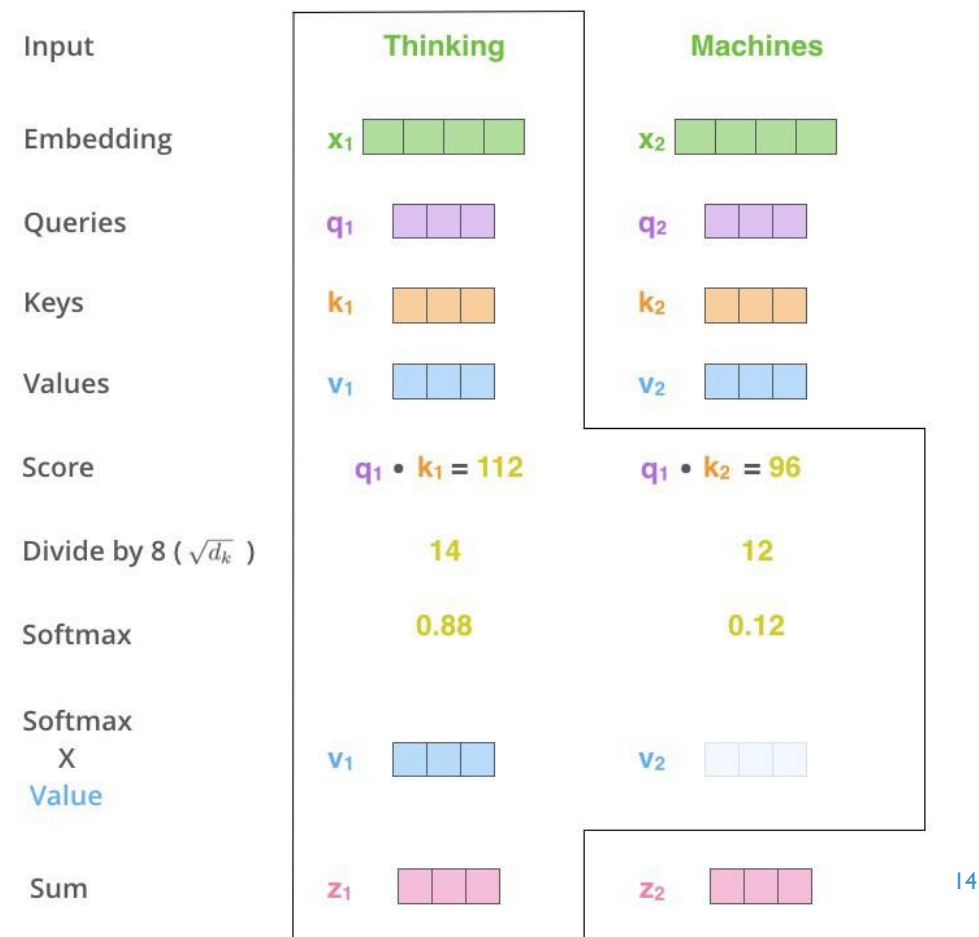
HOW TO CALCULATE SELF ATTENTION

- Step 3:
 - Divide the score by 8
 - Take the square root of the dimension of the key vector
- Step 4:
 - Apply SoftMax to normalizes scores

Input	Thinking	Machines
Embedding	x_1	x_2
Queries	q_1	q_2
Keys	k_1	k_2
Values	v_1	v_2
Score	$q_1 \cdot k_1 = 112$	$q_1 \cdot k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12

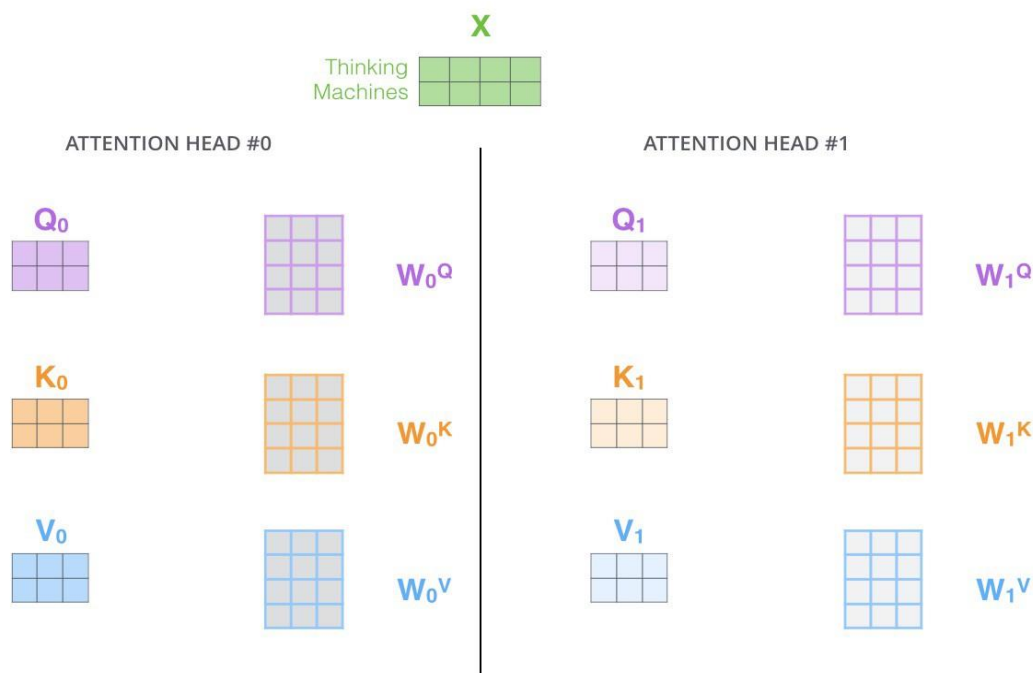
HOW TO CALCULATE SELF ATTENTION

- Step 5:
 - multiply each value vector by the SoftMax score
 - **Intuition:** to amplify relevant words and downgrade irrelevant words
- Step 6:
 - sum up the weighted value vectors.
 - produces the output of the self- attention layer



MULTI-HEADED ATTENTION

- Multi-headed attention improves the performance of the attention layer by allowing for:
 - It expands the model's ability to focus on different positions.
 - It gives the attention layer multiple “representation subspaces”.

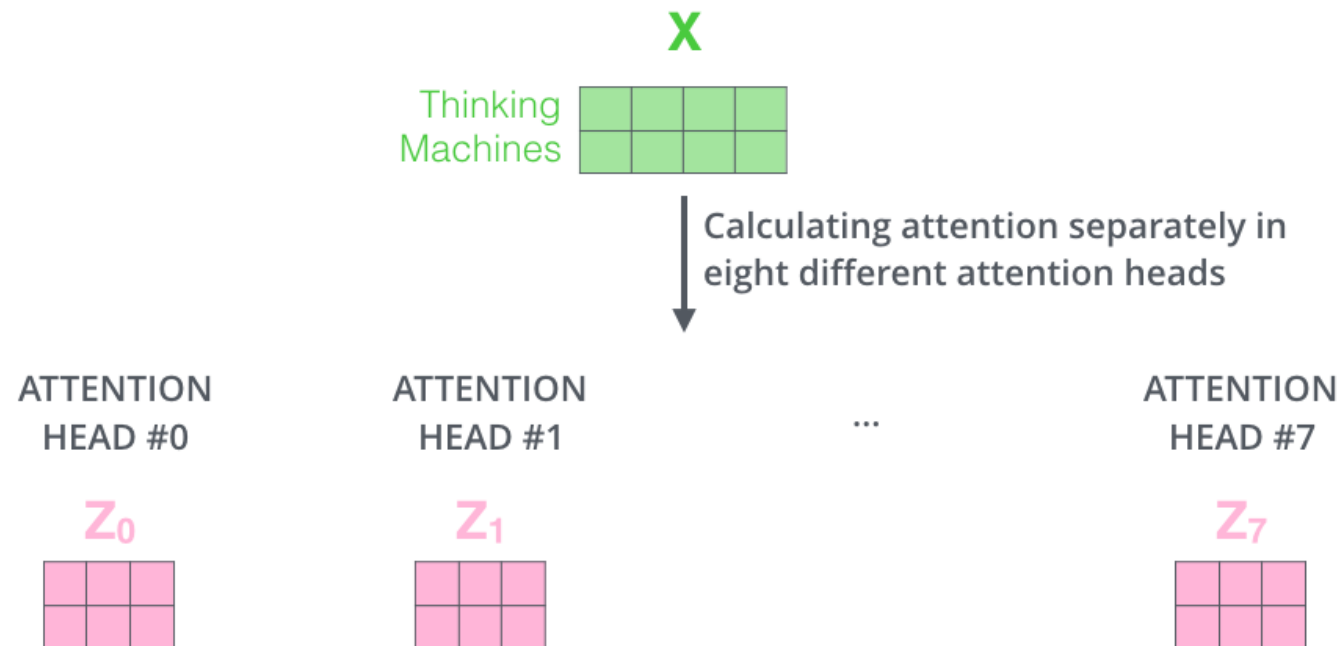


$$\text{softmax}\left(\frac{\mathbf{Q} \times \mathbf{K}^T}{\sqrt{d_k}}\right) \mathbf{V}$$

= **Z**

MULTI-HEADED ATTENTION

- Assume we have multi-headed attention with 8 self attention (eight Z matrices)



MULTI-HEADED ATTENTION

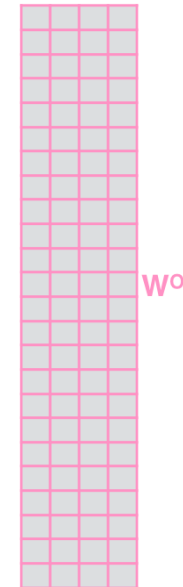
- Assume we have multi-headed attention with 8 self attention (eight Z matrices)
 - the feedforward layer expects a single matrix
 - We concatenate the matrices then multiply them by an additional weight matrix W^O

1) Concatenate all the attention heads

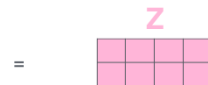


2) Multiply with a weight matrix W^O that was trained jointly with the model

x



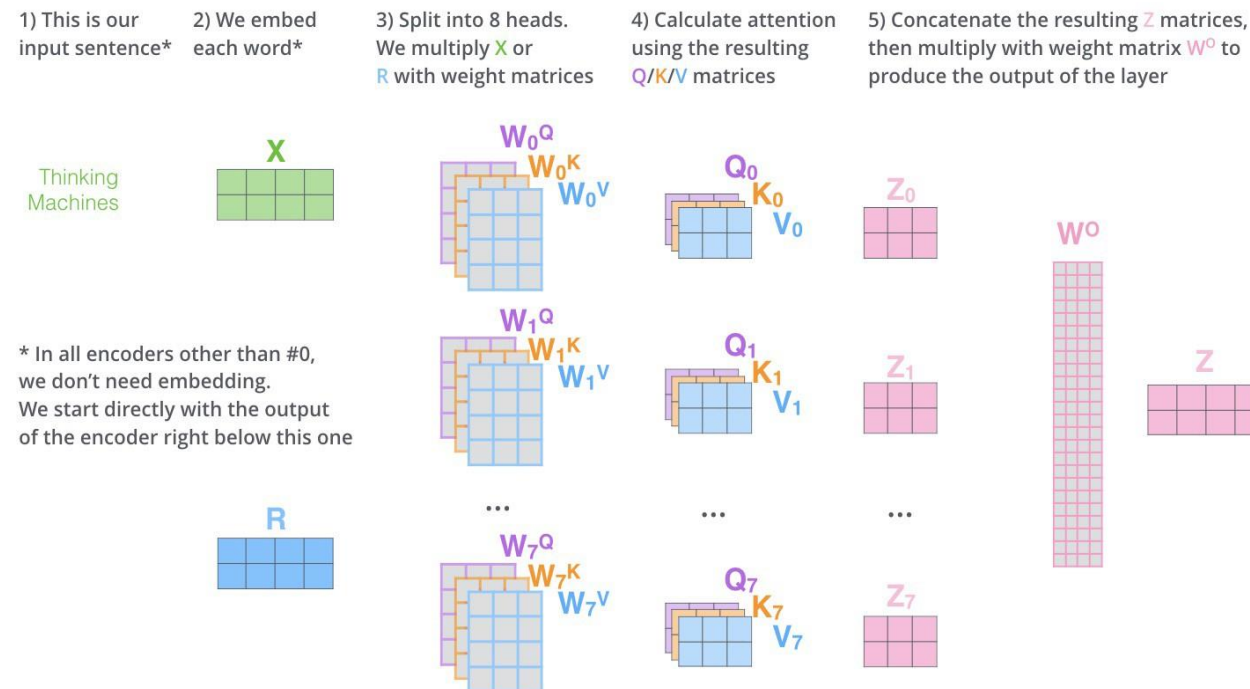
3) The result would be the Z matrix that captures information from all the attention heads. We can send this forward to the FFNN



MULTI-HEADED ATTENTION

Assume we have multi-headed attention with 8 self attention (eight Z matrices)

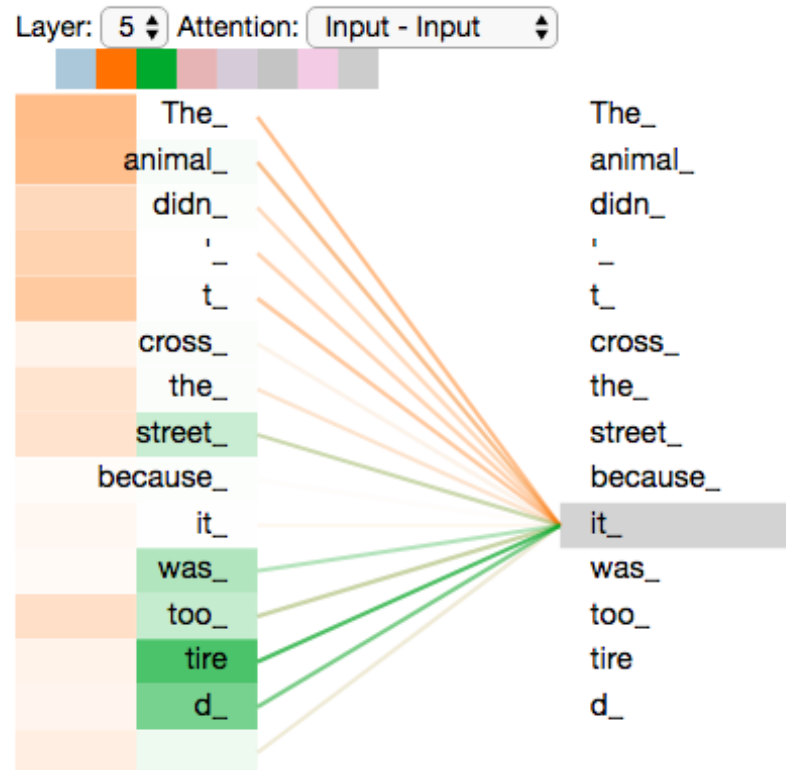
- Concatenate all 8 the feedforward layer expects a single matrix



MULTI-HEADED ATTENTION

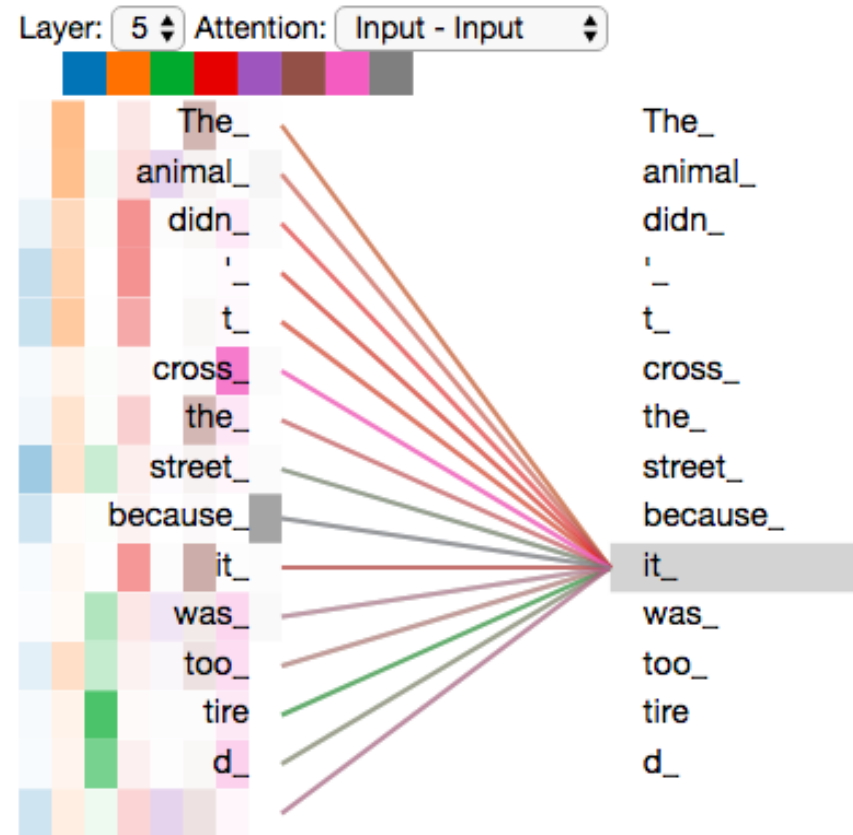
Multi head attention *it*

- one attention head is focusing most on *the animal*
- another is focusing on *tired*



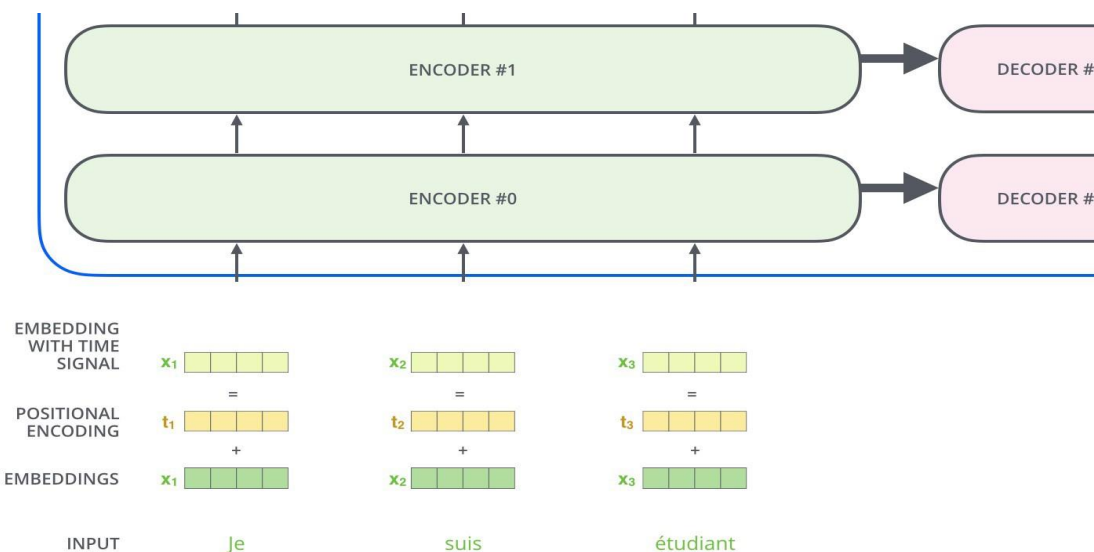
MULTI-HEADED ATTENTION

Visualizing all 8 attention heads



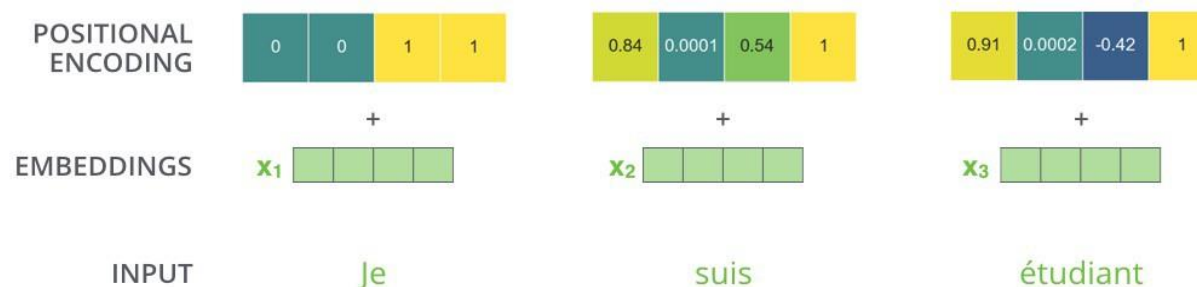
POSITIONAL ENCODING IN TRANSFORMERS

- Positional encoding describes the position of a token in a sequence.
- Each position is assigned a unique representation.
- Each position is mapped to a vector.
- The output of the positional encoding layer is a matrix, where each row of the matrix represents an encoded object of the sequence summed with its positional information.



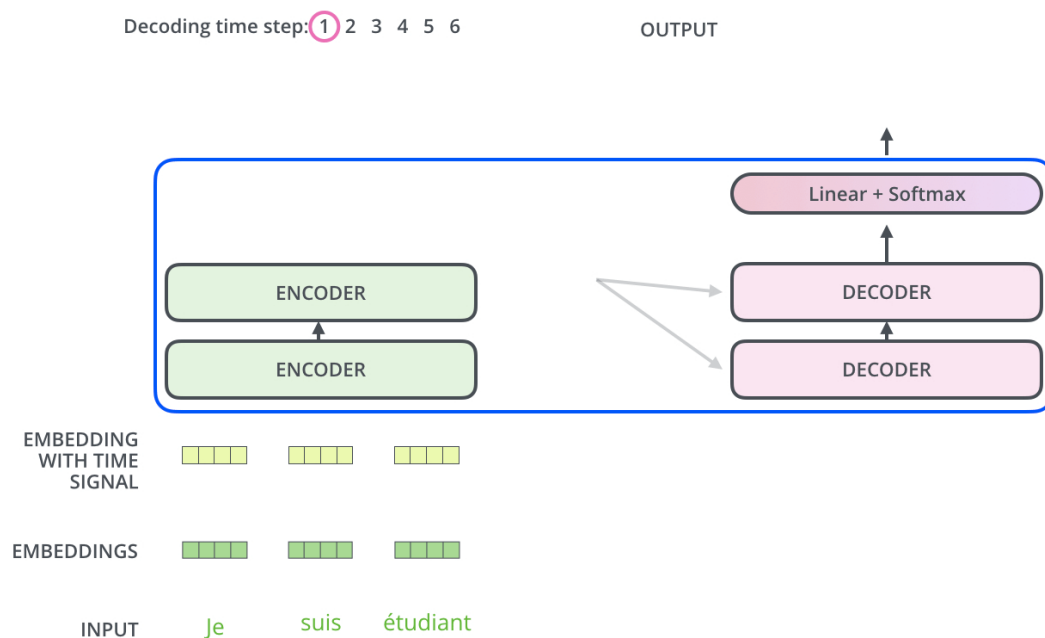
POSITIONAL ENCODING IN TRANSFORMERS

- **Intuition:**
 - Adding these values to the embeddings provides meaningful distances between the embedding vectors once they are projected into Q/K/V vectors and during dot-product attention.
 - These positional encoding are generated during training (detail in the paper)
- Assume encoding of dimension 4, the positioning will look like



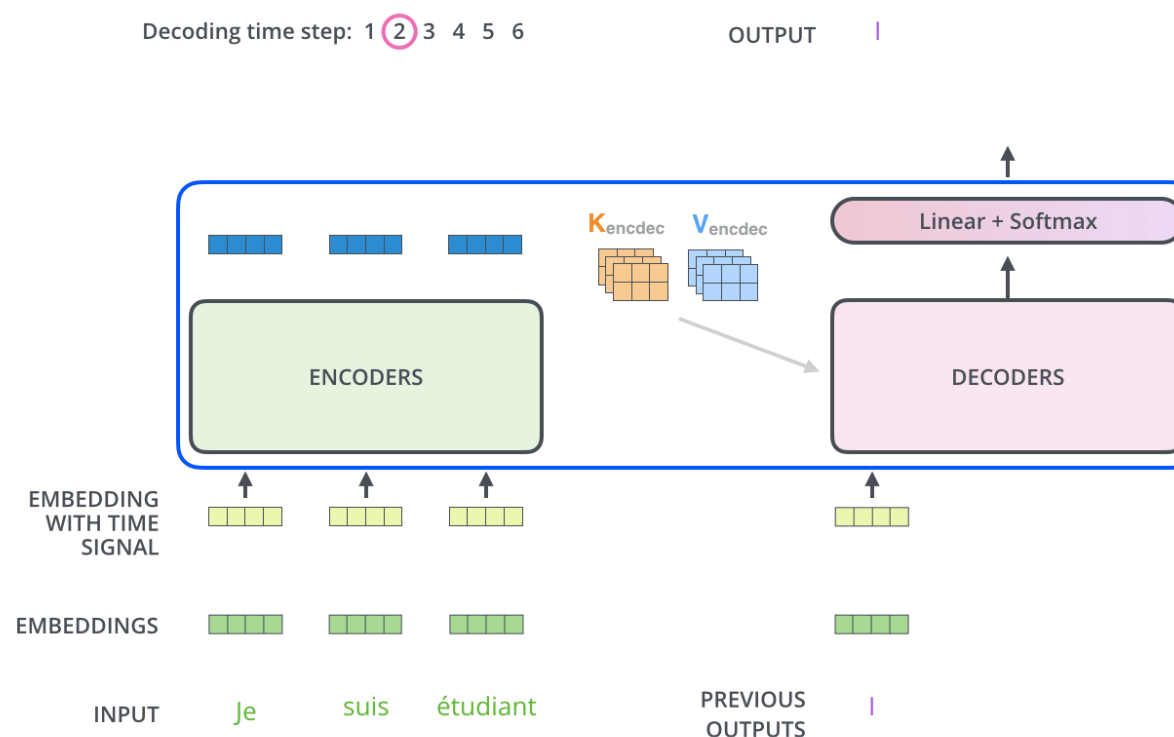
THE DECODER SIDE

- The encoder process the input sentence
- Output of top encoder layer is converted to attention vectors K and V.
- K & V vectors are used by each decoder in its “encoder-decoder attention” layer
- K & V helps decoder focus on appropriate words.



THE DECODER SIDE

The following steps repeat the process until a special symbol is reached indicating the transformer decoder has completed its output

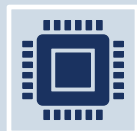


THE DECODER SIDE



The self attention layers is only allowed to focus on earlier positions in the output sequence.

This is done by masking future positions (setting them to $-\infty$) before the SoftMax step in the self-attention calculation.



The “Encoder-Decoder Attention” layer is like multiheaded self-attention

except it creates its Queries matrix from the layer below it, takes the Keys and Values matrix from the output of the encoder stack.

FINAL LINEAR AND SOFTMAX LAYER

Final Linear Layer:

- fully connected neural network that projects the vector produced by the decoders into a logits vector

Assume our model was trained on a 1000-word vocabulary

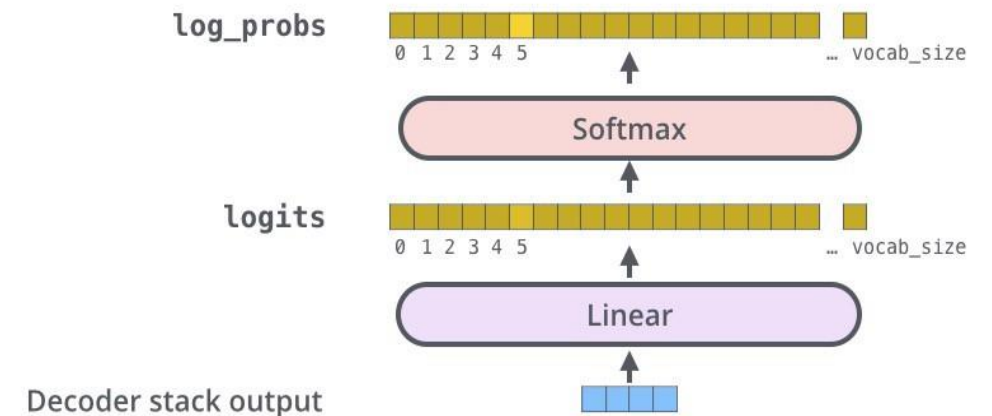
- Then logit vector has dimension 1000 cells
- Each cell the score of a unique word
- The SoftMax layers turns these scores into probabilities and the word with the highest probability is chosen.

Which word in our vocabulary
is associated with this index?

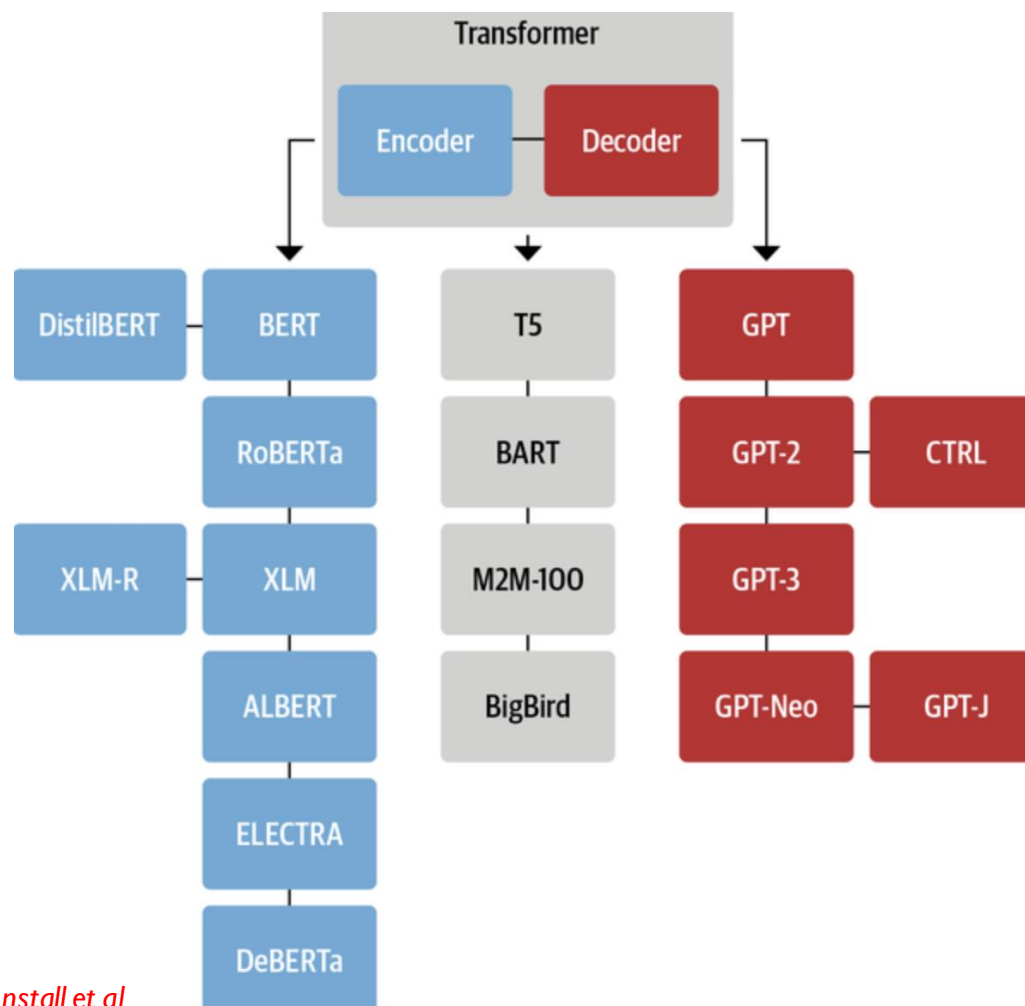
am

Get the index of the cell
with the highest value
(argmax)

5

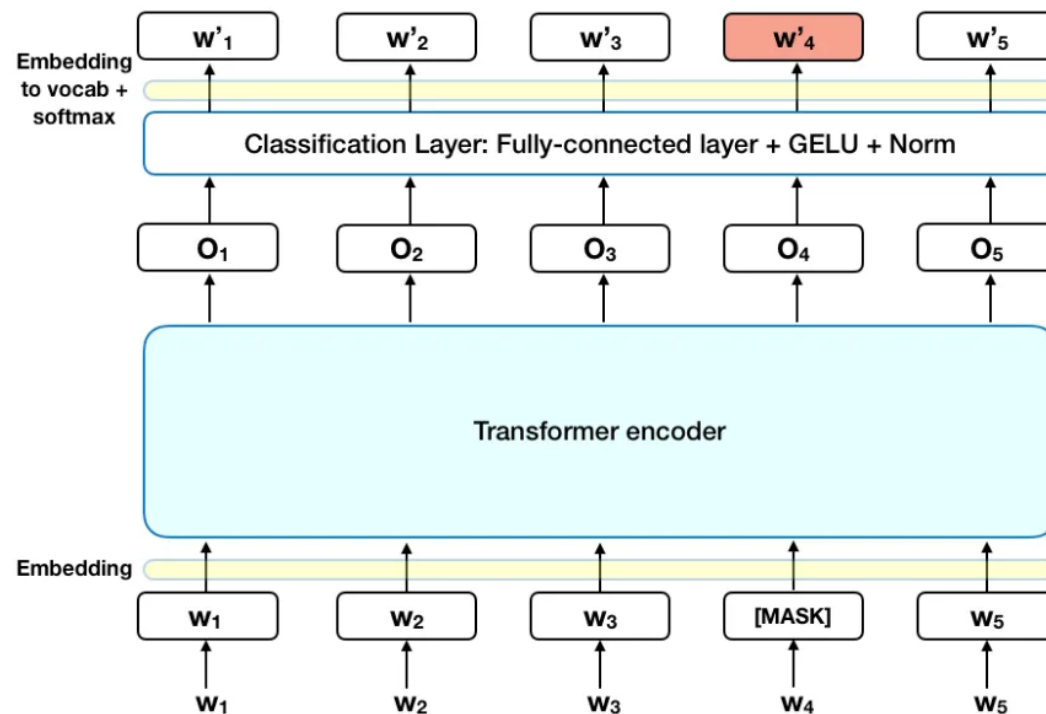


OVERVIEW OF MOST PROMINENT TRANSFORMERS

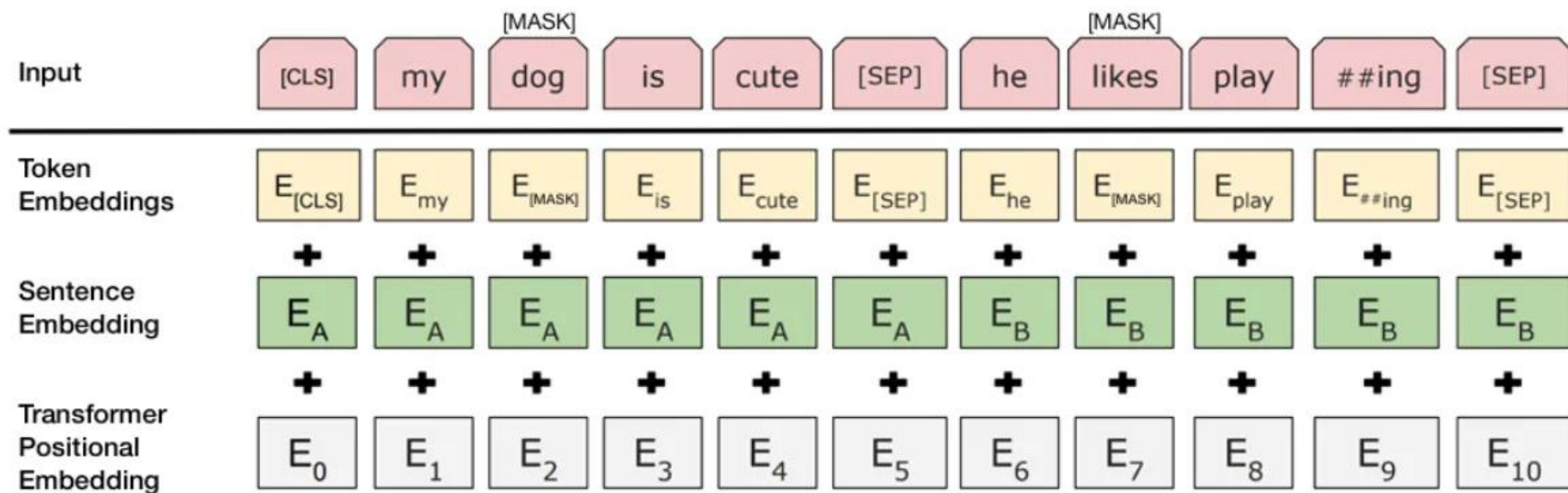


OVERVIEW OF MOST PROMINENT TRANSFORMERS – BERT

Dual training of Masked language model and next sentence prediction.
Masked language model



BERT – NEXT SENTENCE PREDICTION

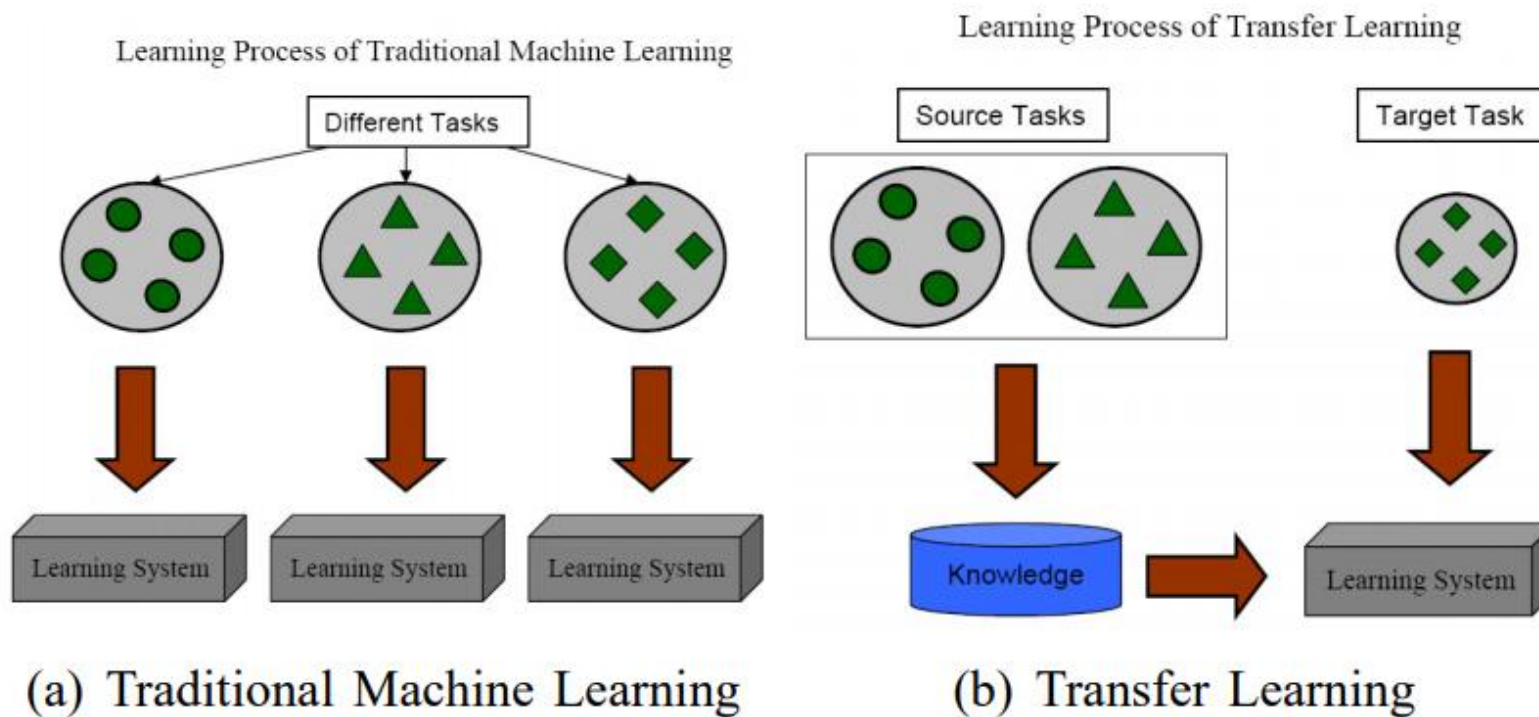




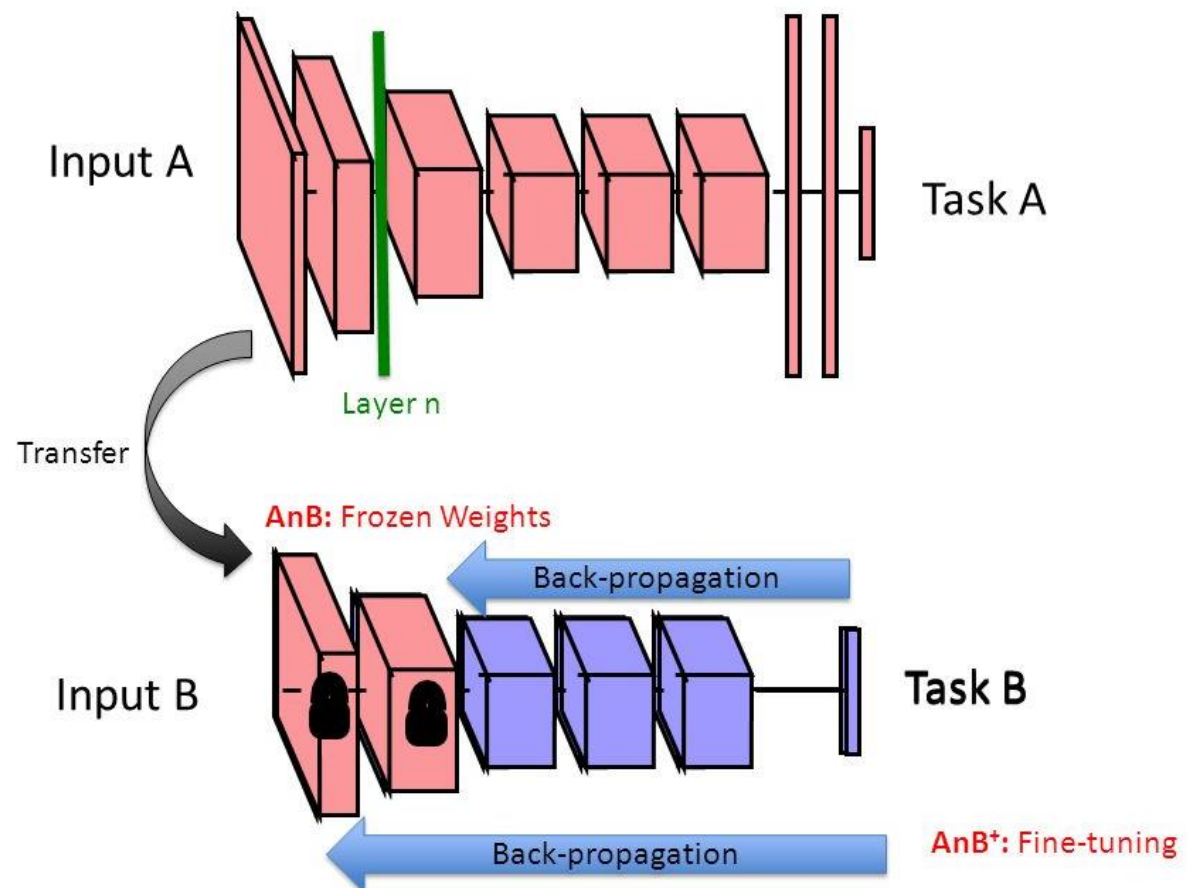
TRANSFER LEARNING

TRANSFER LEARNING

Transferring the knowledge of one model to perform a new task.



TRANSFER LEARNING



TRANSFER LEARNING

Different types of transfer learning

Type	Description	Examples
Inductive	Adapt existing supervised training model on new labeled dataset	Classification, Regression
Transductive	Adapt existing supervised training model on new unlabeled dataset	Classification, Regression
Unsupervised	Adapt existing unsupervised training model on new unlabeled dataset	Clustering, Dimensionality Reduction

USEFUL LINKS

- Attention is all you need https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
- BERT <https://aclanthology.org/N19-1423.pdf>
- Transfer learning <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.147.9185&rep=rep1&type=pdf>
- Hugging face
 - <https://huggingface.co/transformers/index.html>
 - <https://github.com/huggingface/transformers>
 - <https://huggingface.co/docs/transformers/training>
 - <https://neptune.ai/blog/hugging-face-pre-trained-models-find-the-best>