

NLP Advanced - 01

PRE-TRAINED LANGUAGE MODEL

AI VIET NAM
Nguyen Quoc Thai

CONTENT

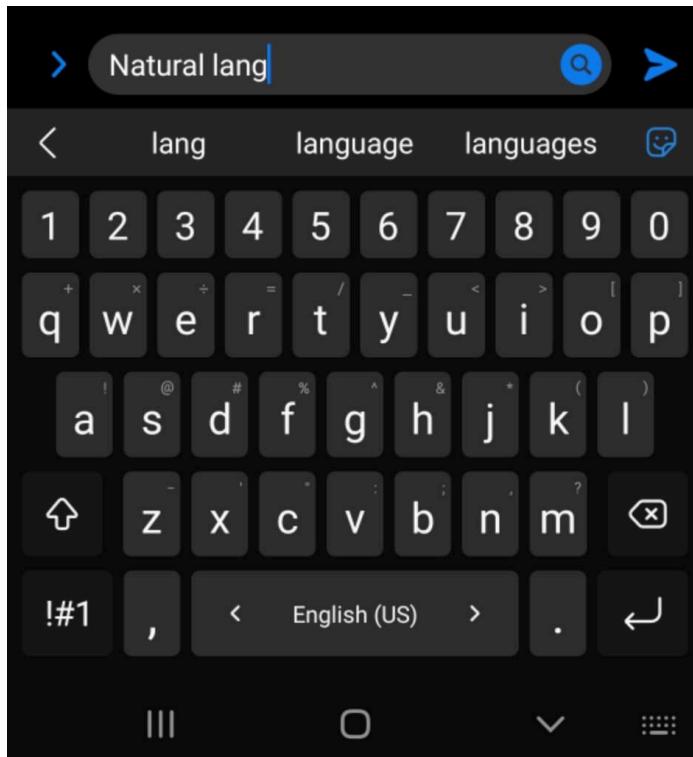
- 1 **Language Model**
- 2 **Transformer Architecture**
- 3 **Pre-trained Language Model**
- 4 **Applications**

1 – Language Model

!

Language Model

- The task of predicting what word comes next



1 – Language Model

!

Language Model

- Assign a probability $P(w_1, w_2, \dots, w_n)$ to every finite sequence w_1, w_2, \dots, w_n
- Predict a next token based on history tokens

Completion

I am learning Natural Language ...

Processing

 $P(\text{Processing} | \text{I am learning Natural Language})$

Conditional Probability

 $P(\text{Handling} | \text{I am learning Natural Language})$

Handling

1 – Language Model

!

Statistical Language Model

- The probability of a word depends only on some previous words

$$P(w_{1:n}) = \prod_{i=1}^n P(w_i | w_{i-N+1:i-1})$$

$$P(w_i | w_{1:i-1}) = P(w_i | w_{i-N+1:i-1})$$

- N-gram model with $N = \{1, 2, \dots\}$

1 – Language Model

!

Statistical Language Model

➤ Compute $P(w|h)$

$$P(w|h) = P(\text{learning}|I, \text{am})$$

w: token as word “learning”

h: history tokens as “I,am”

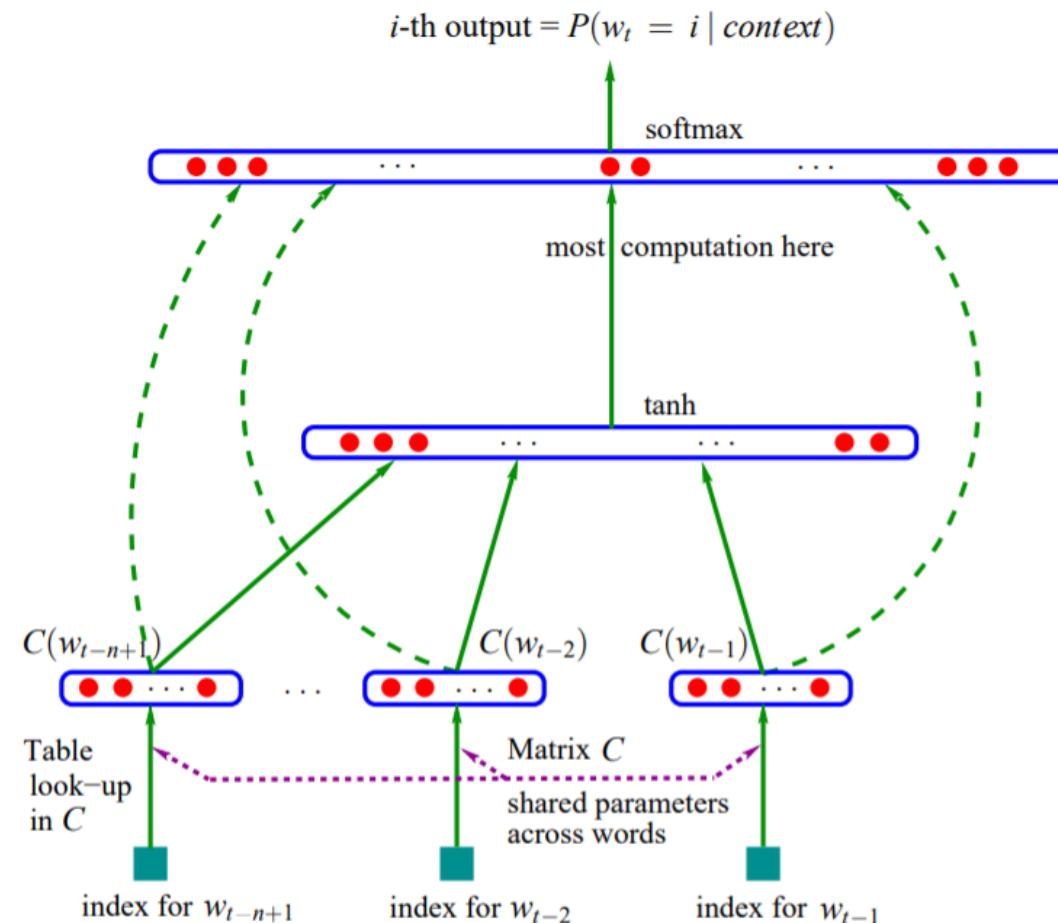
$$P(w|h) = \frac{\text{count}(hw)}{\text{count}(h)}$$

$$P(\text{school}|i, \text{go}, \text{to}) = \frac{\text{count}(I, \text{am}, \text{learning})}{\text{count}(I, \text{am})}$$

1 – Language Model



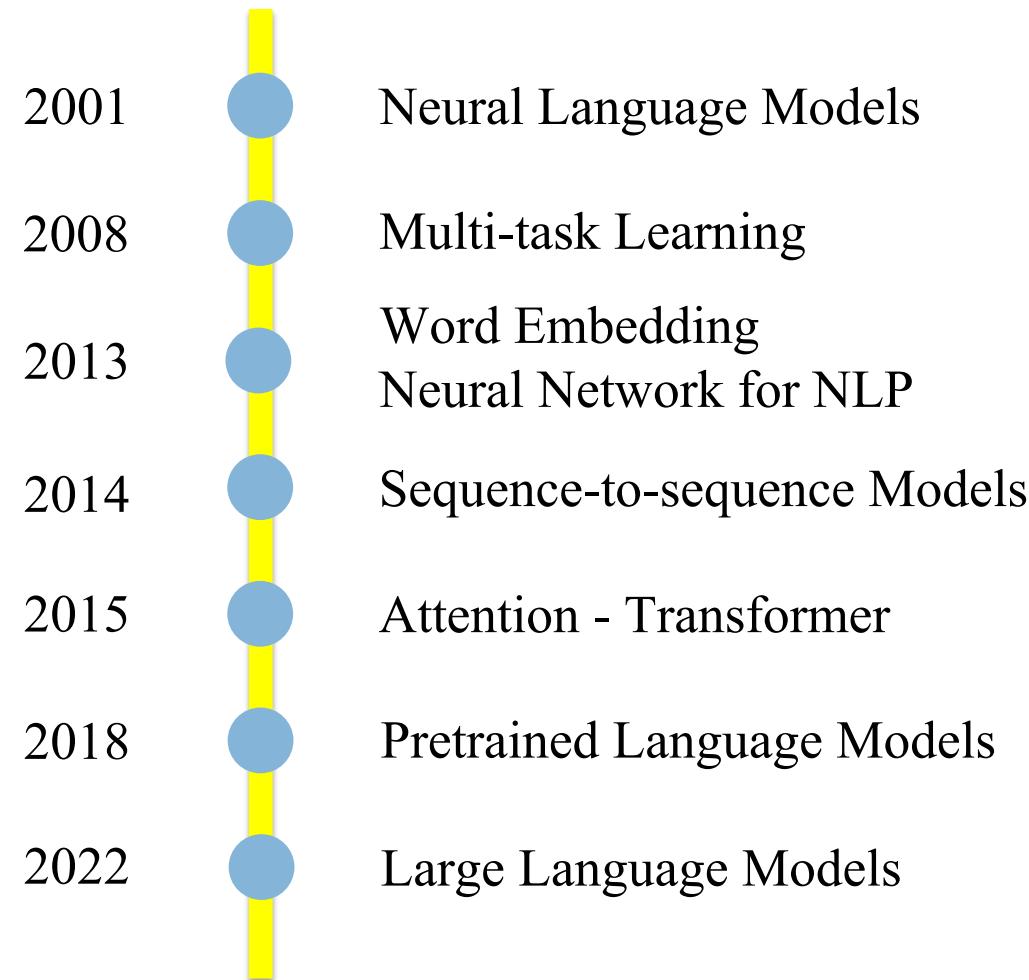
Neural Language Model



1 – Language Model

!

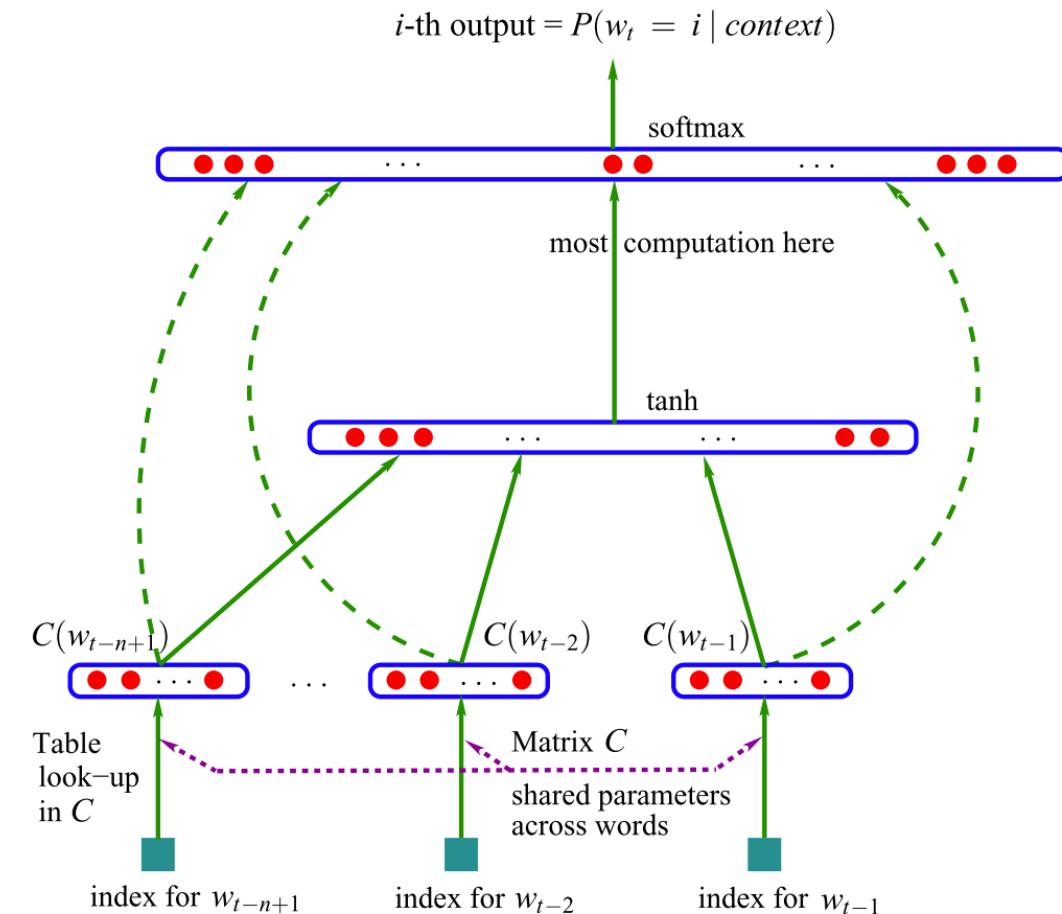
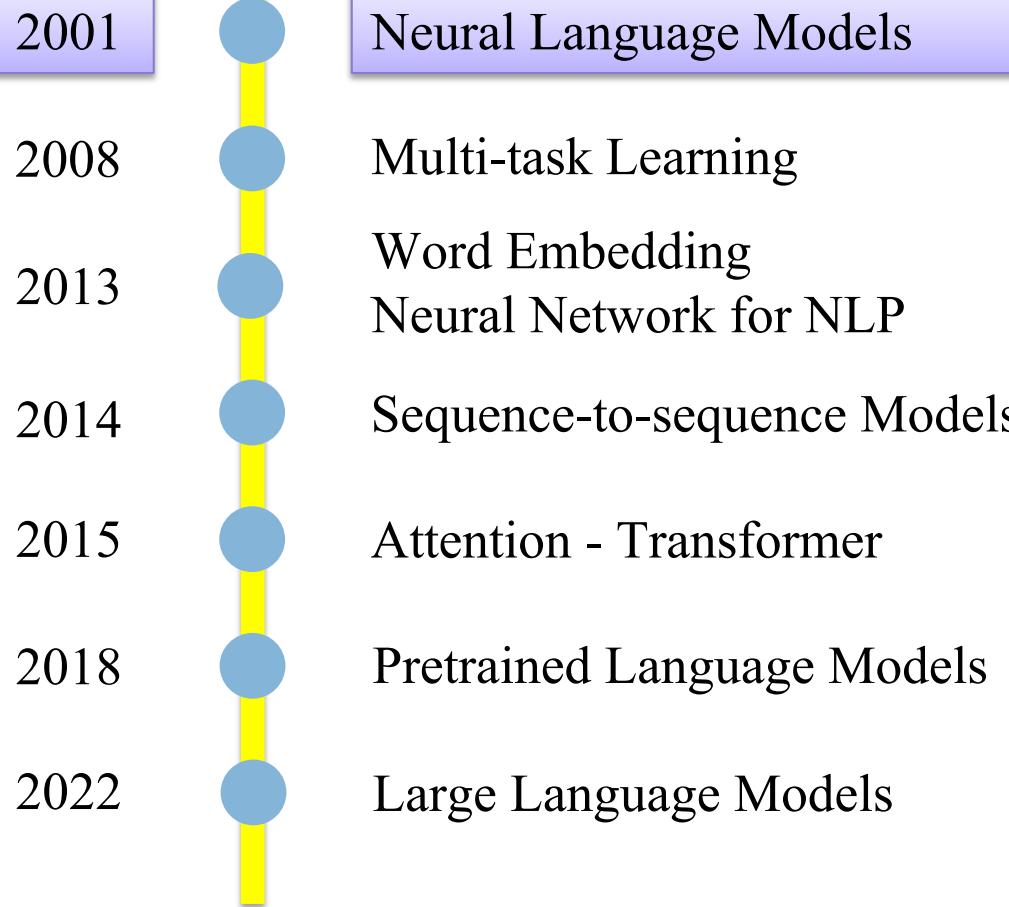
The neural history of NLP



1 – Language Model



The neural history of NLP

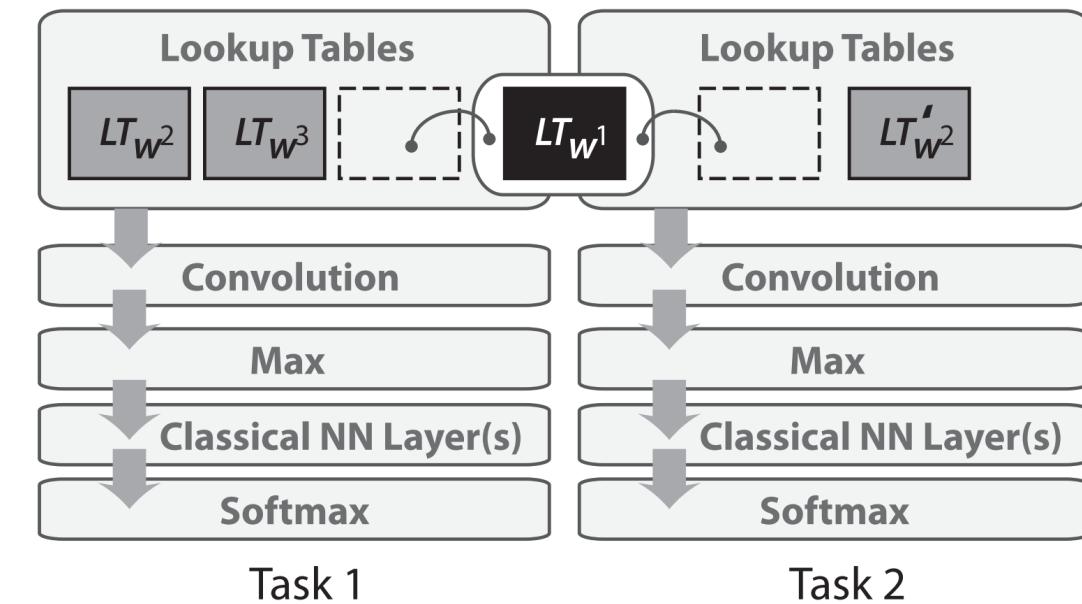


1 – Language Model



The neural history of NLP

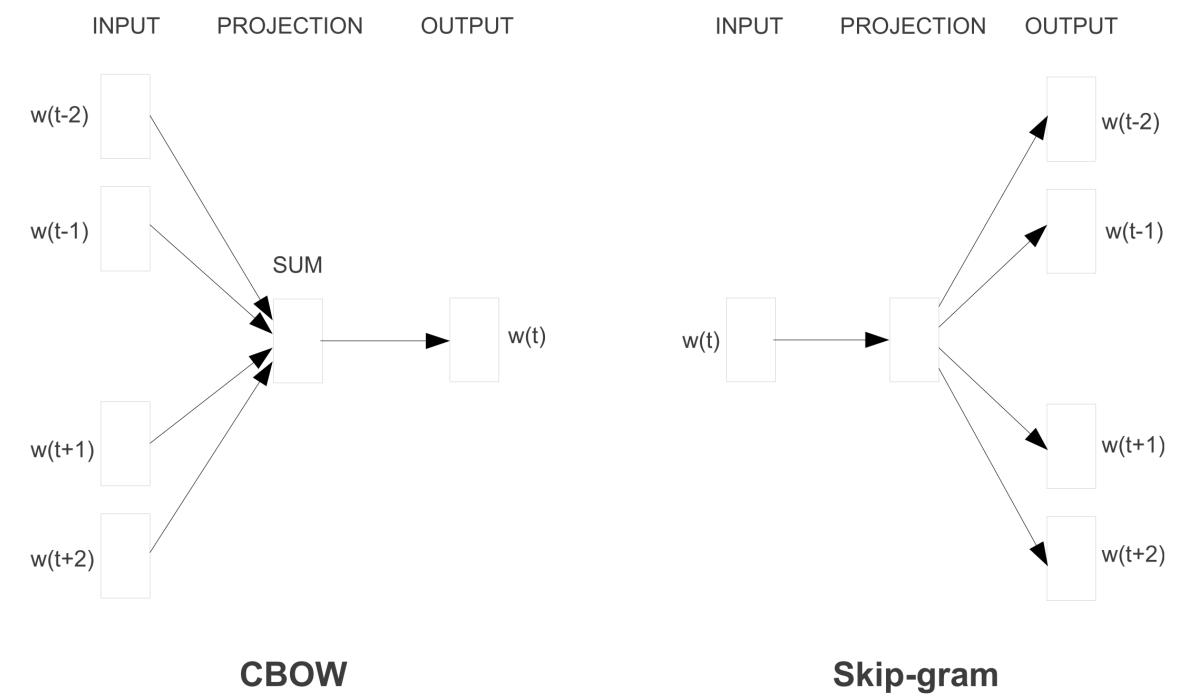
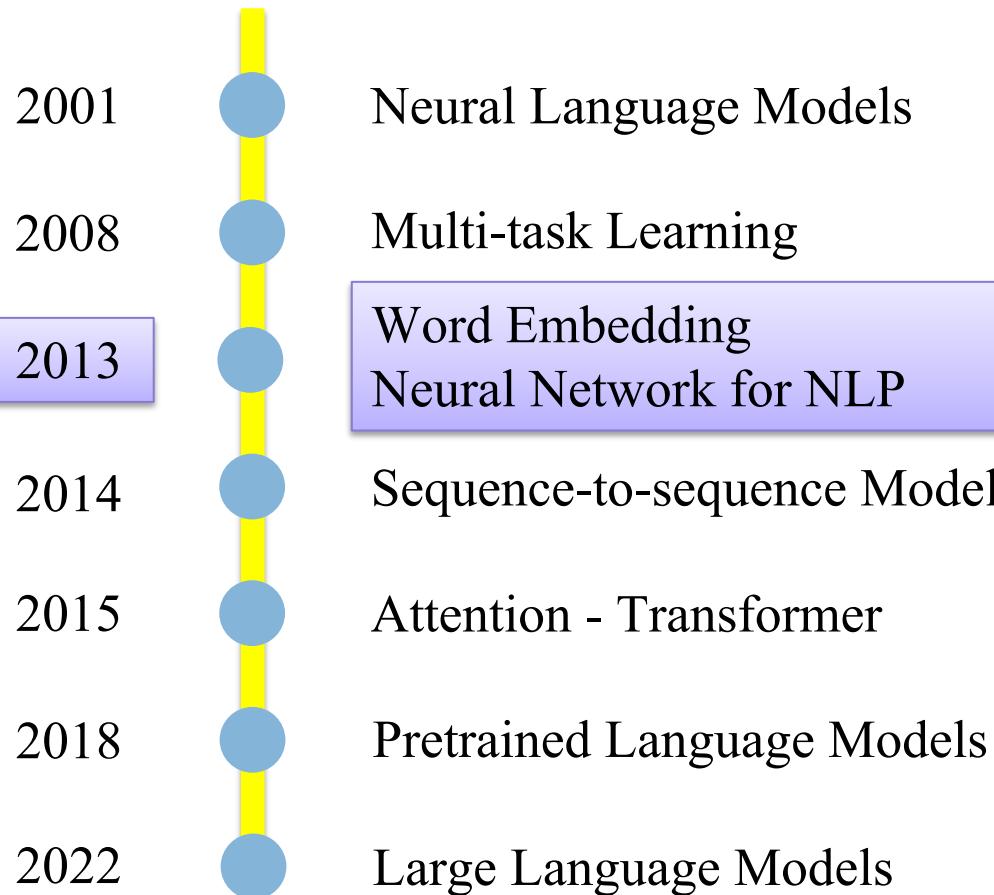
2001	Neural Language Models
2008	Multi-task Learning
2013	Word Embedding Neural Network for NLP
2014	Sequence-to-sequence Models
2015	Attention - Transformer
2018	Pretrained Language Models
2022	Large Language Models



1 – Language Model



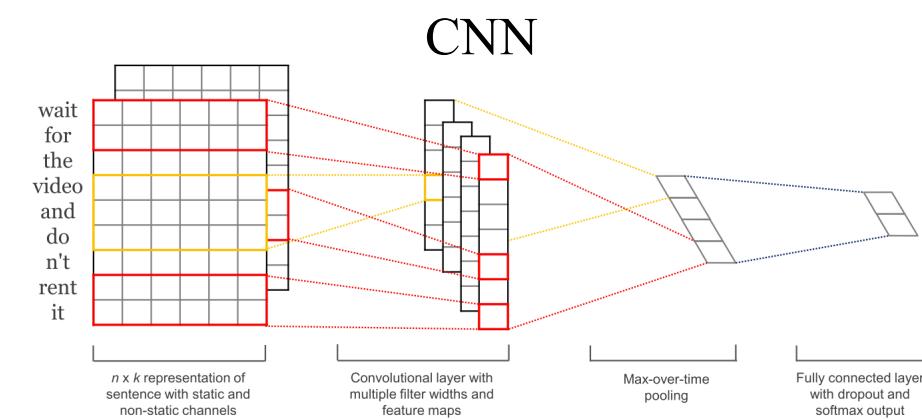
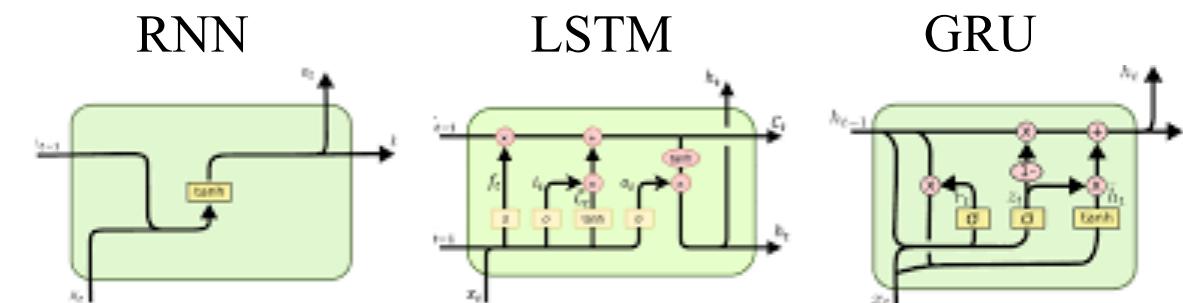
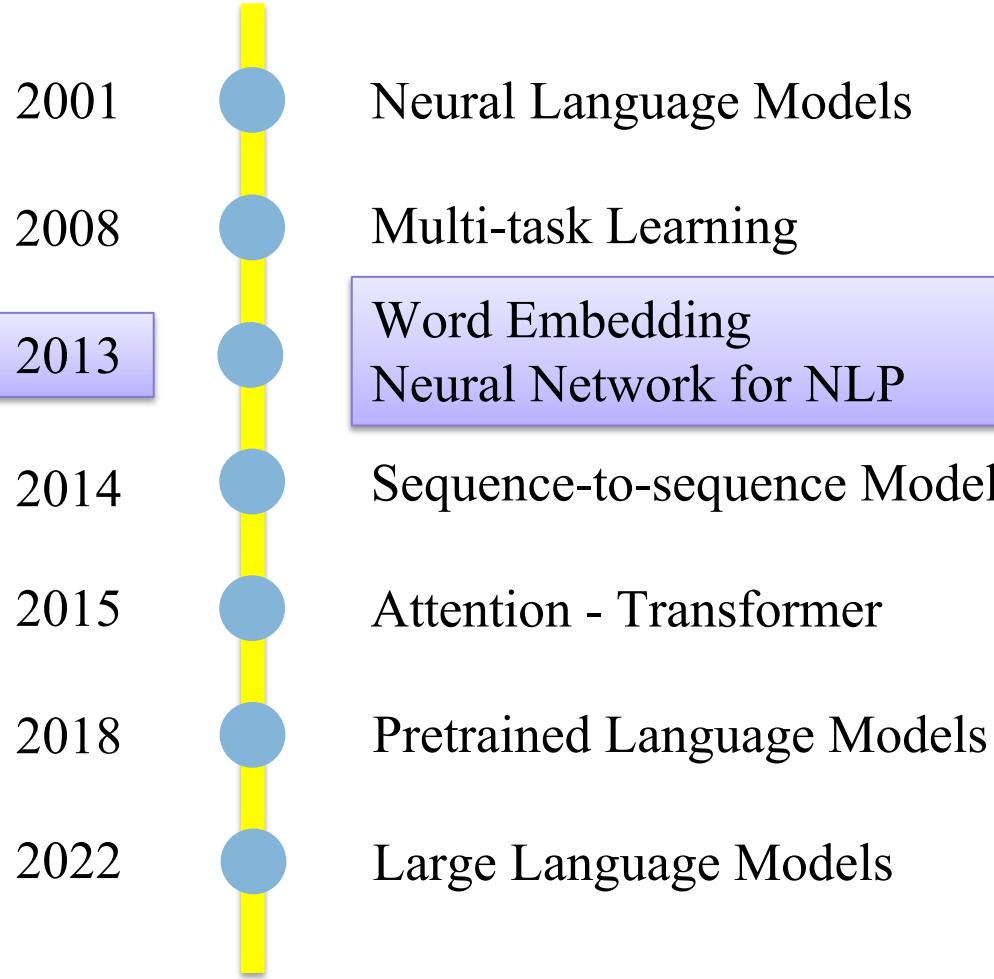
The neural history of NLP



1 – Language Model



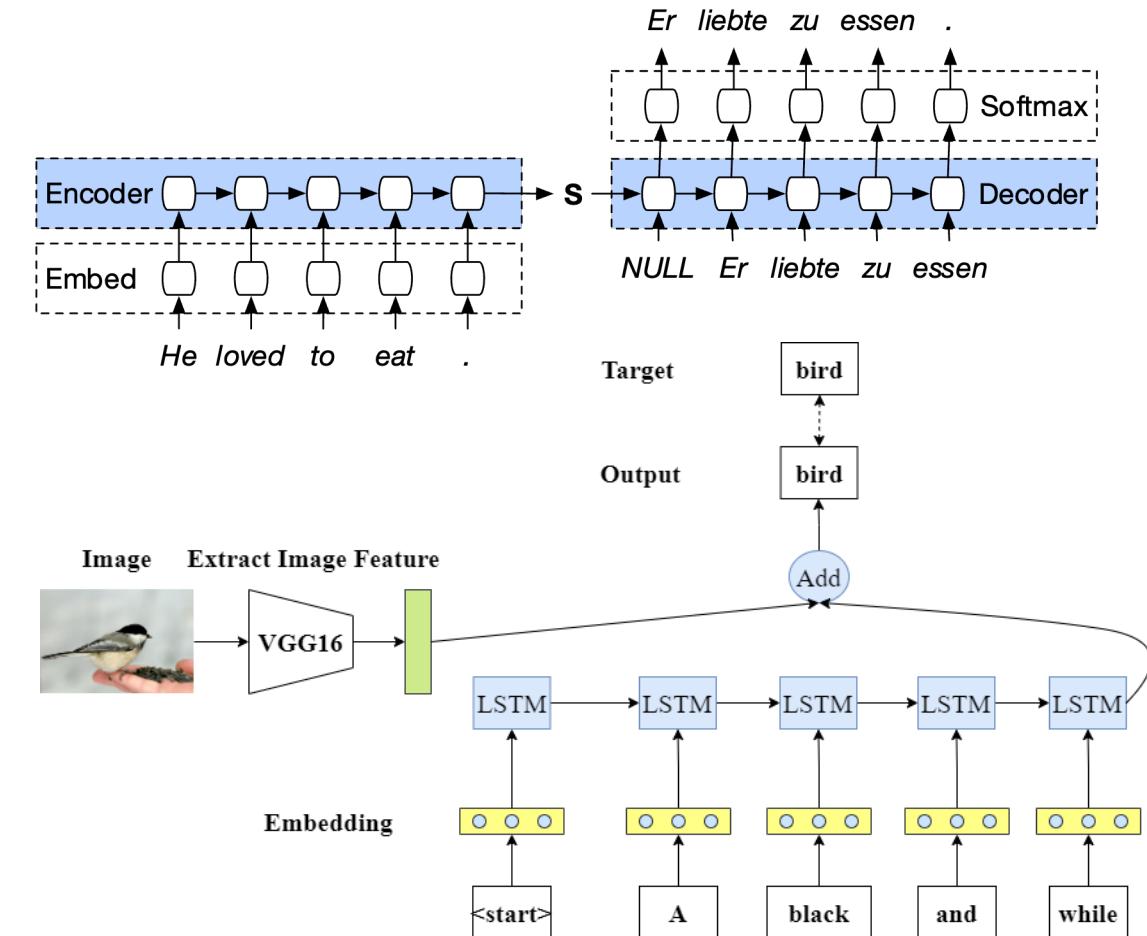
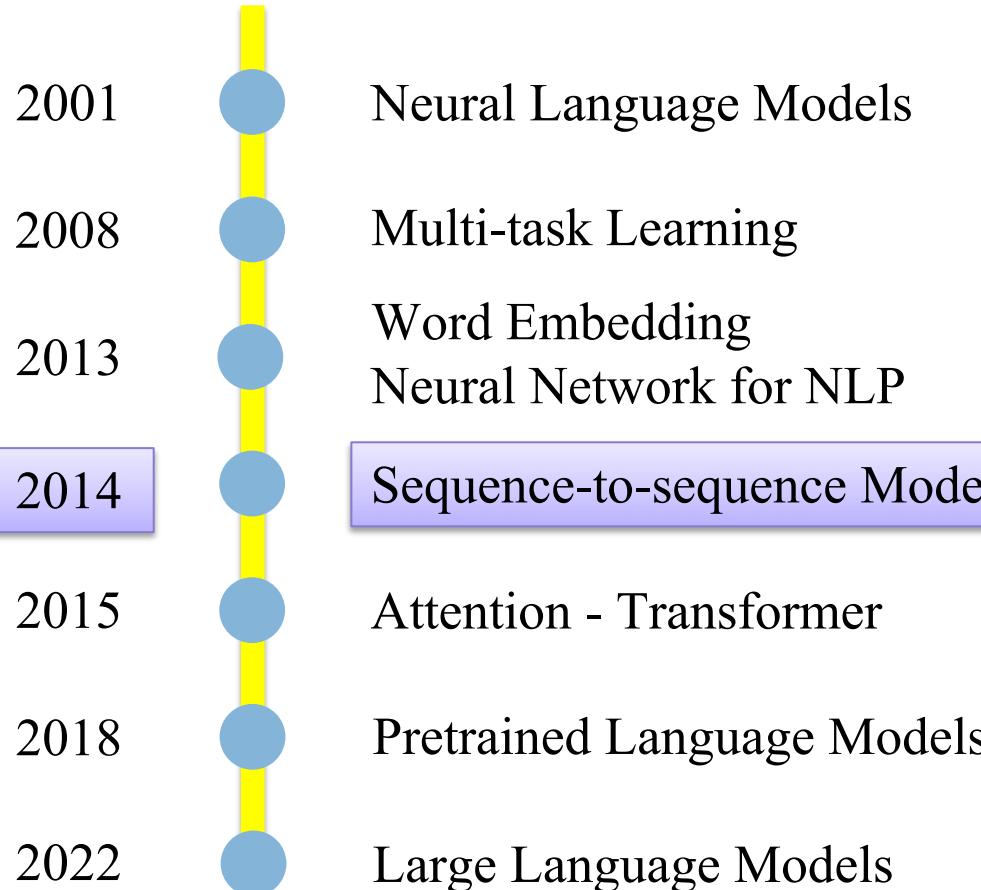
The neural history of NLP



1 – Language Model

!

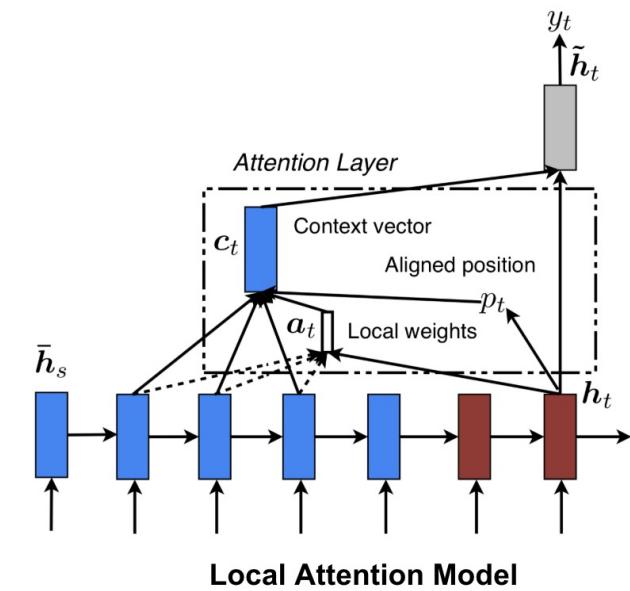
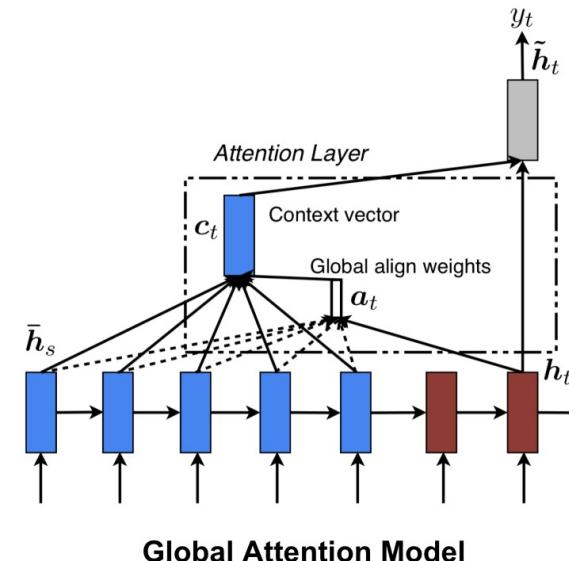
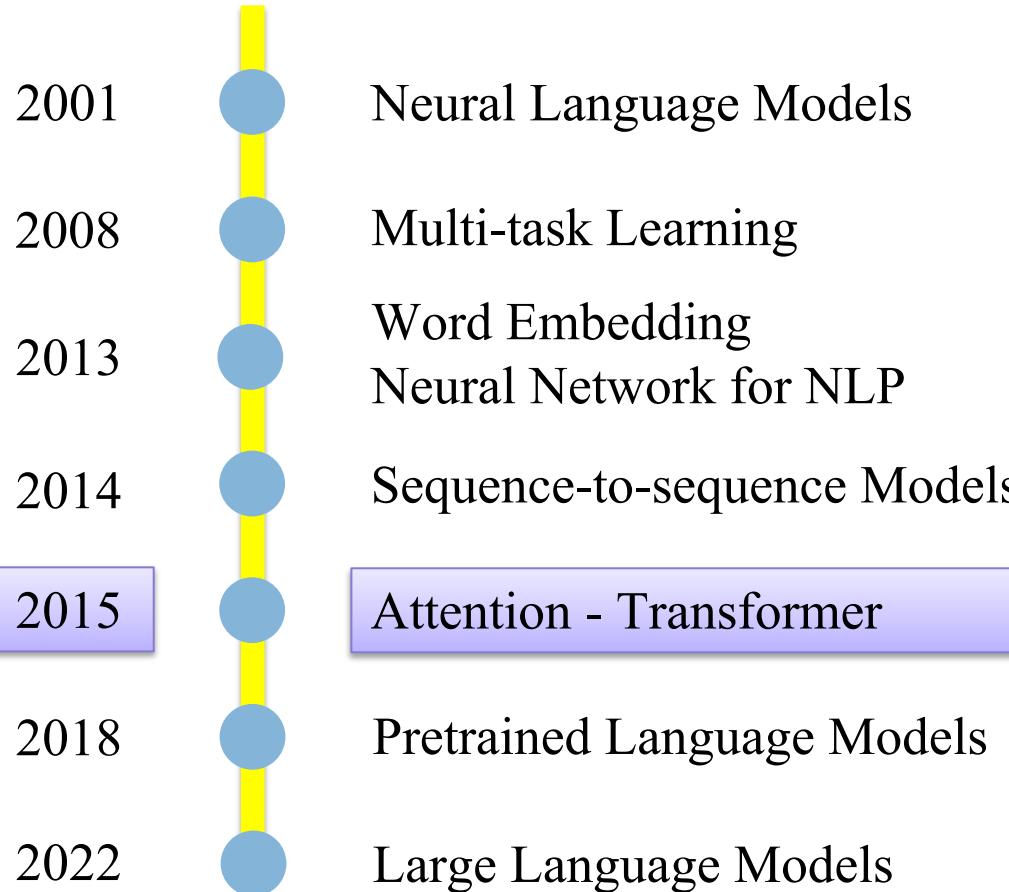
The neural history of NLP



1 – Language Model



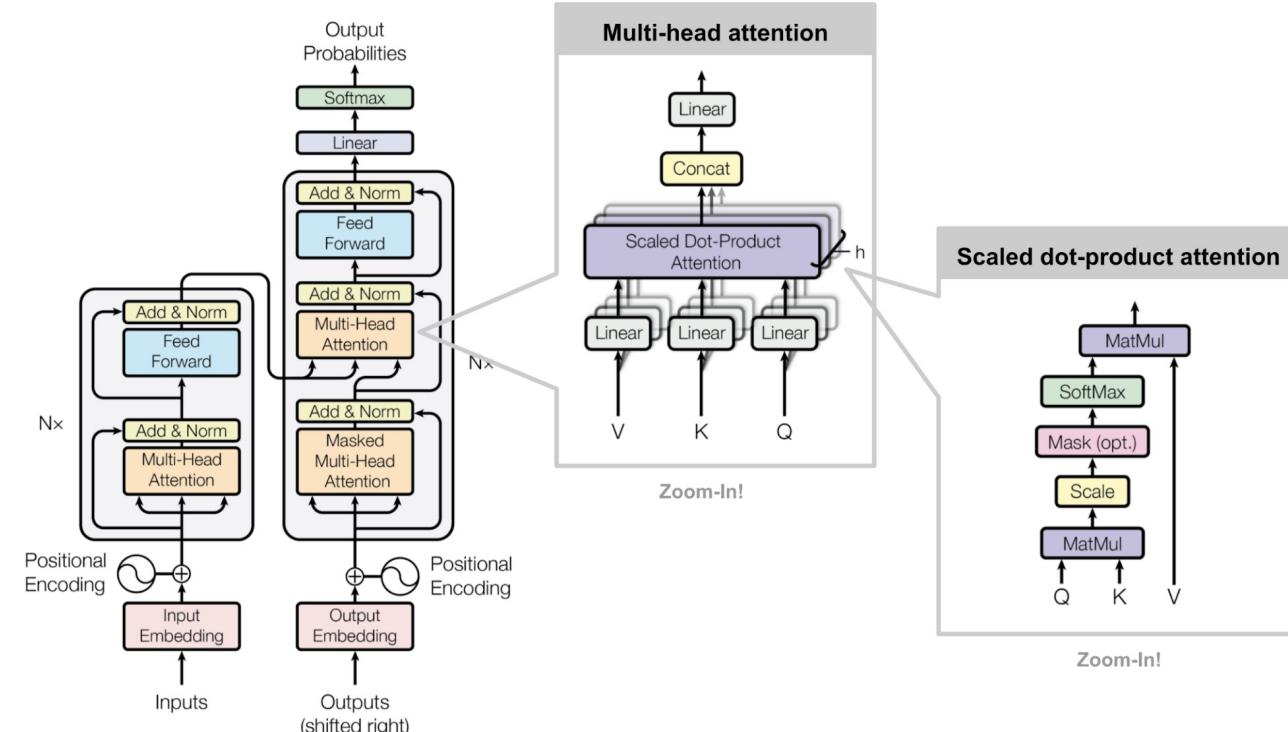
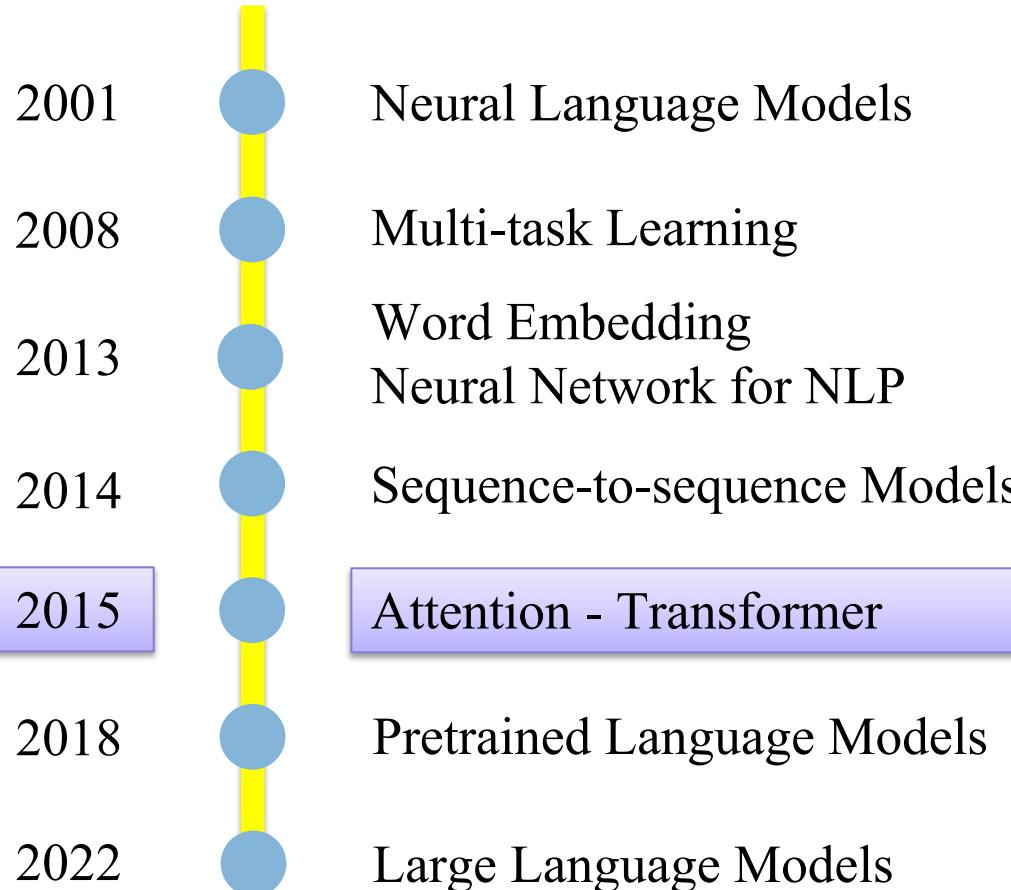
The neural history of NLP



1 – Language Model



The neural history of NLP

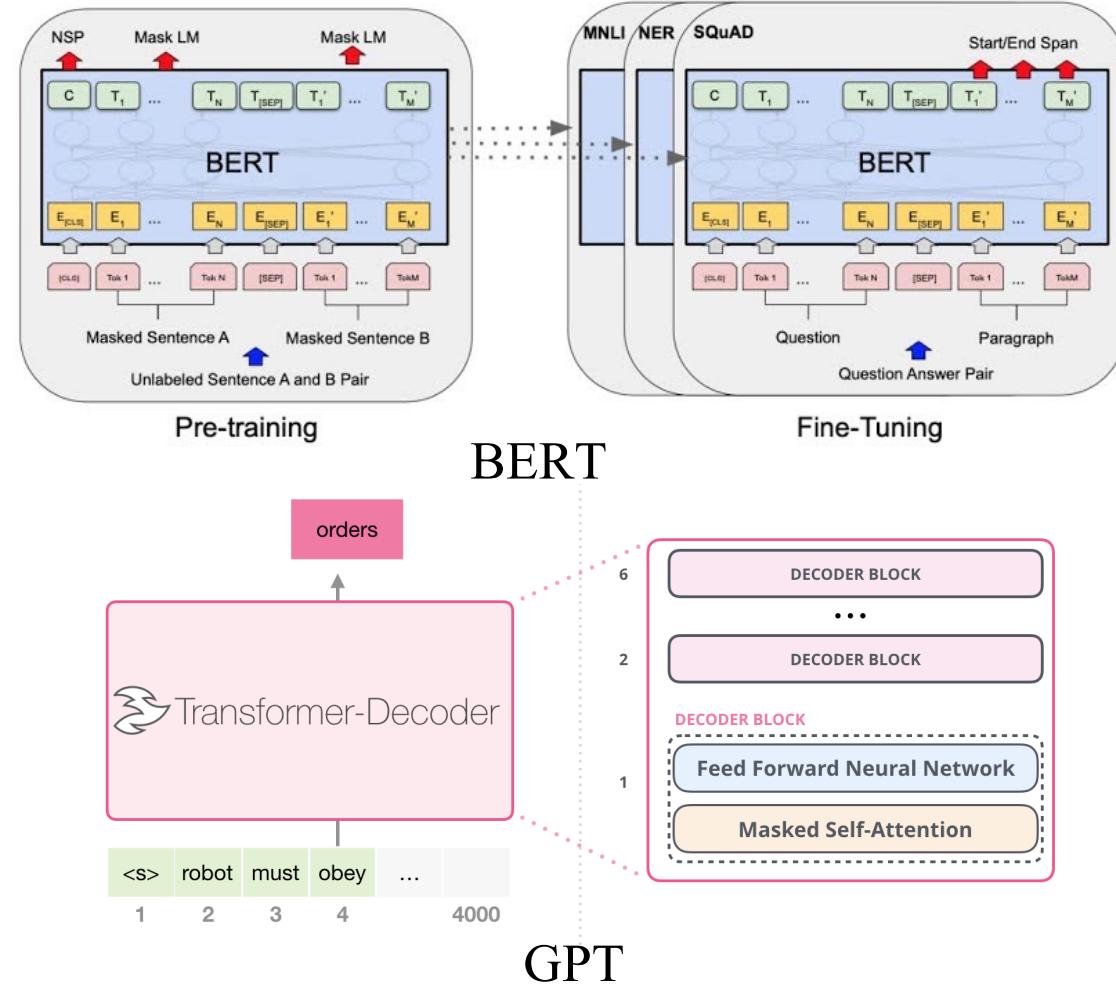


1 – Language Model



The neural history of NLP

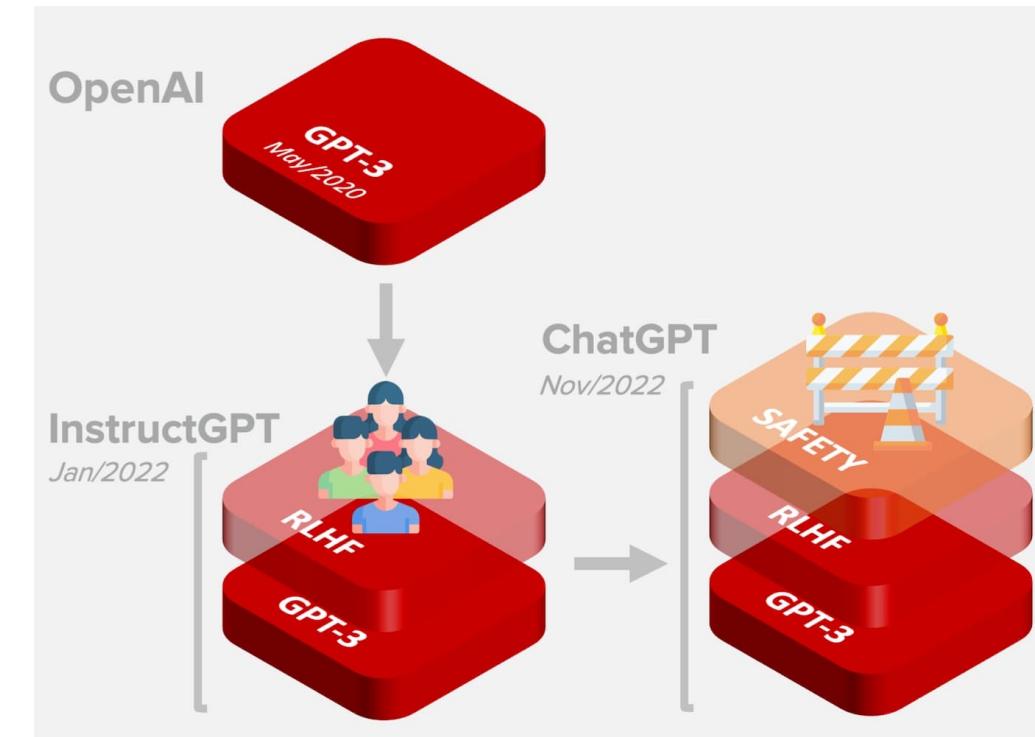
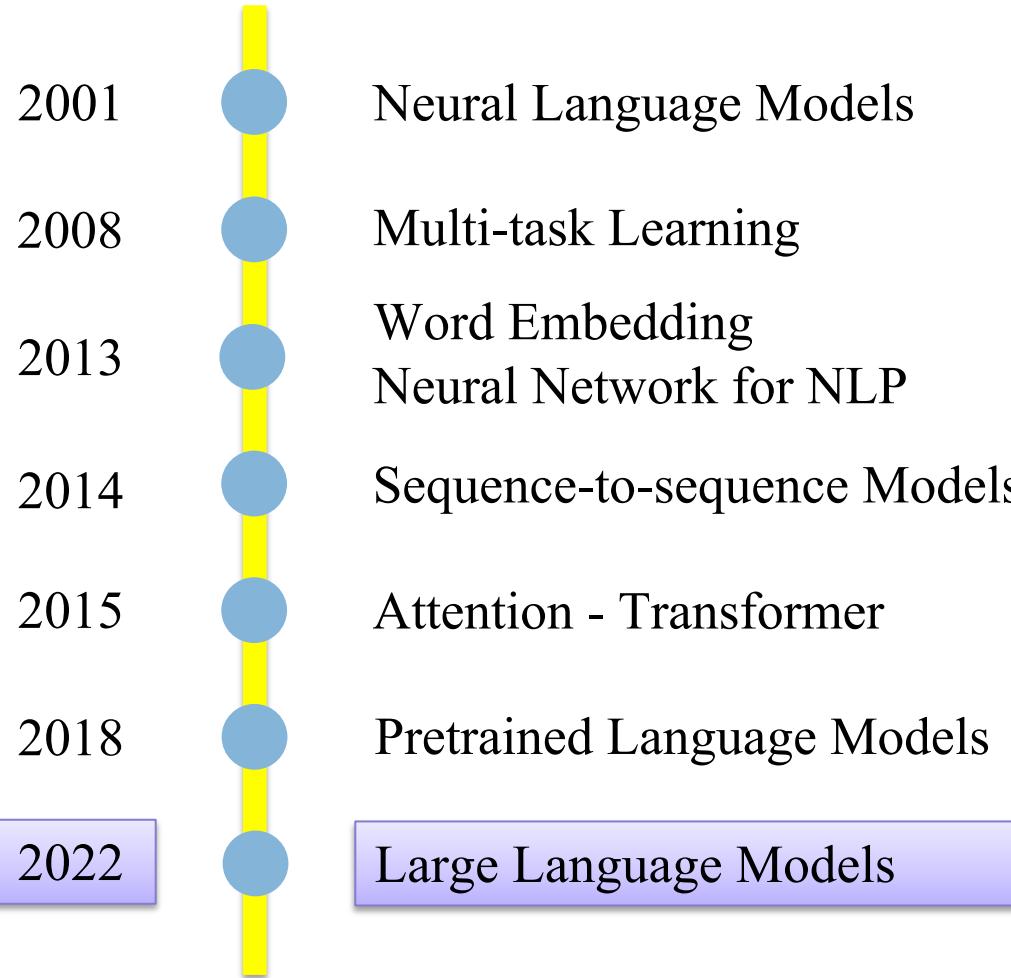
2001	Neural Language Models
2008	Multi-task Learning
2013	Word Embedding Neural Network for NLP
2014	Sequence-to-sequence Models
2015	Attention - Transformer
2018	Pretrained Language Models
2022	Large Language Models



1 – Language Model

!

The neural history of NLP



2 – Transformer

!

Machine Translation Task

- Translate a sentence $w^{(s)}$ in a **source language (input)** to a sentence $w^{(t)}$ in the **target language (output)**



2 – Transformer



Machine Translation Task

- Translate a sentence $w^{(s)}$ in a **source language (input)** to a sentence $w^{(t)}$ in the **target language (output)**
- Can be formulated as an optimization problem:

$$\hat{w}^{(t)} = \operatorname{argmax}_{w^{(t)}} \theta(w^{(s)}, w^{(t)})$$

Where θ is a scoring function over source and target sentences

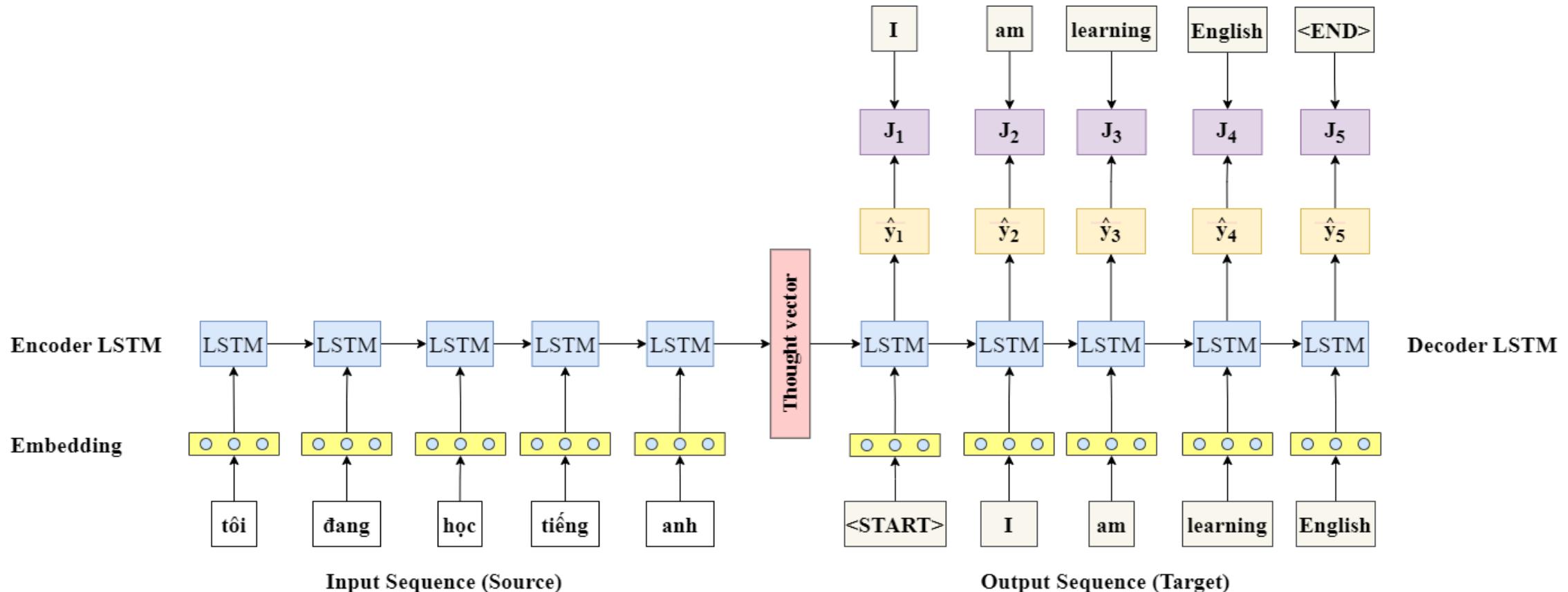
- Requires two components:
 - Learning algorithm to compute parameters of θ
 - Decoding algorithm for computing the best translation $\hat{w}^{(t)}$

2 – Transformer



Machine Translation Task

- Machine Translation using Sequence to Sequence (Seq2Seq) architecture

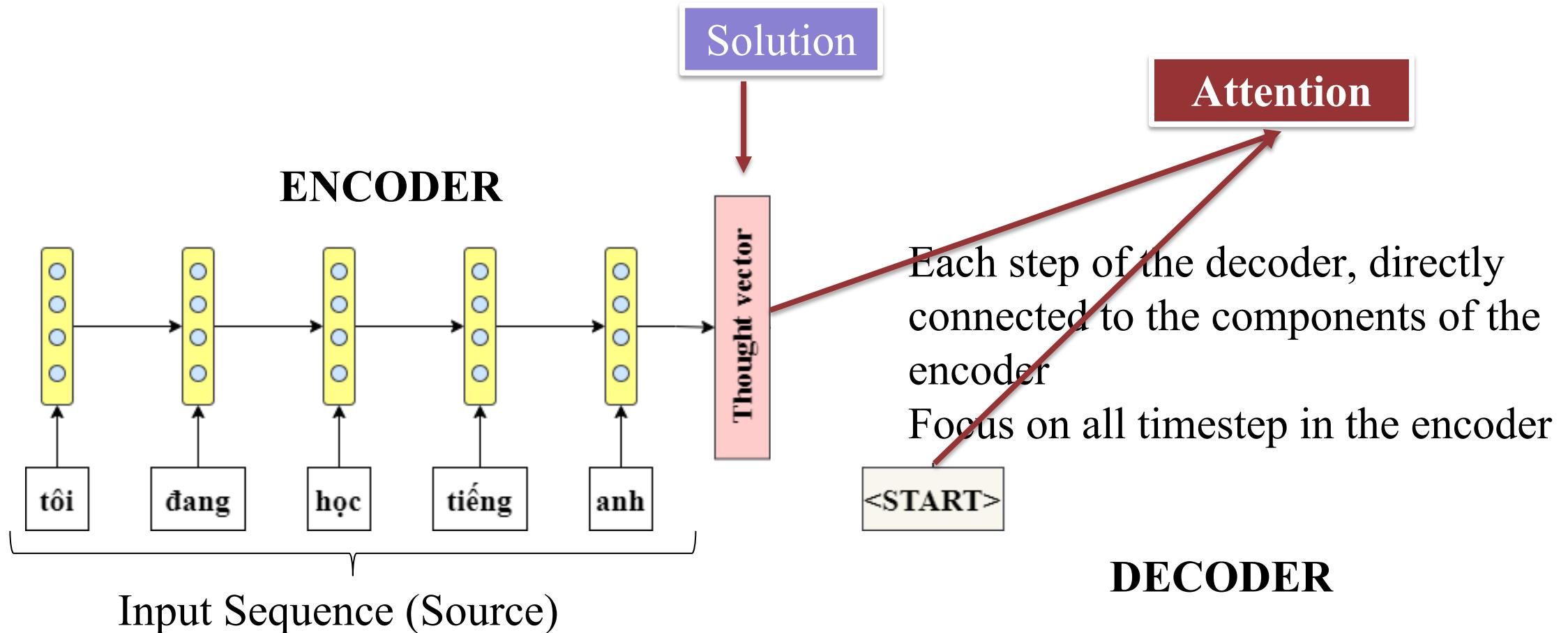


2 – Transformer



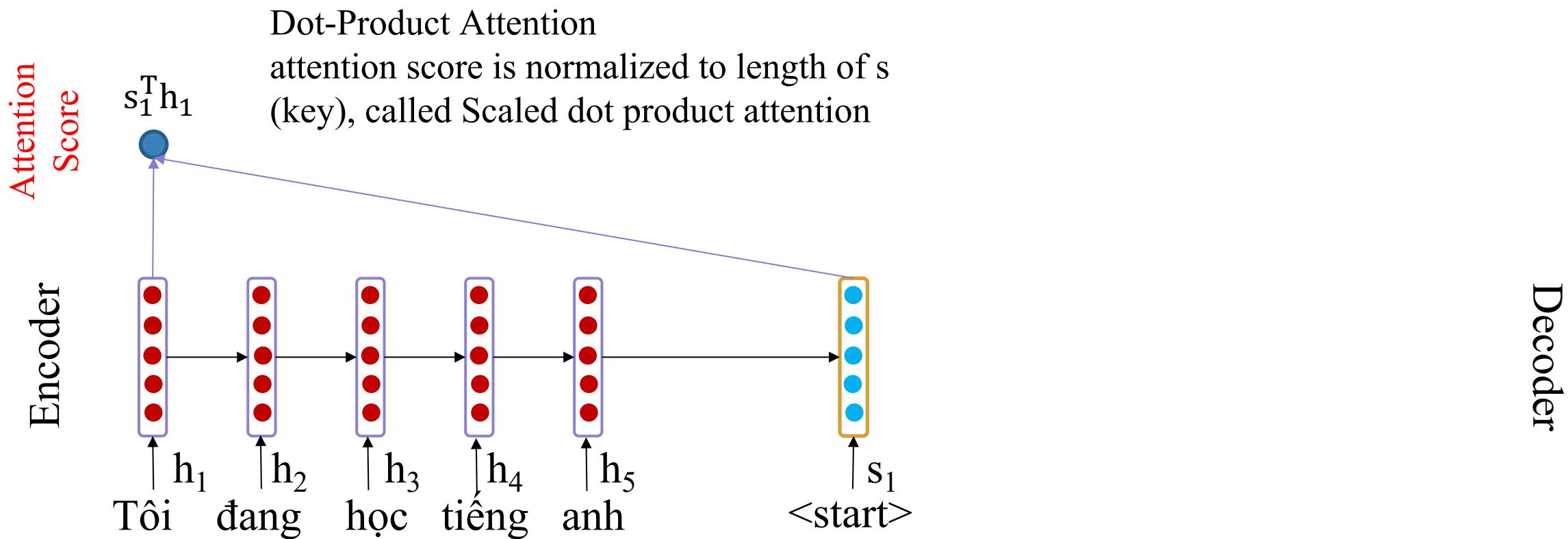
Machine Translation Task

- Machine Translation using Sequence to Sequence (Seq2Seq) architecture





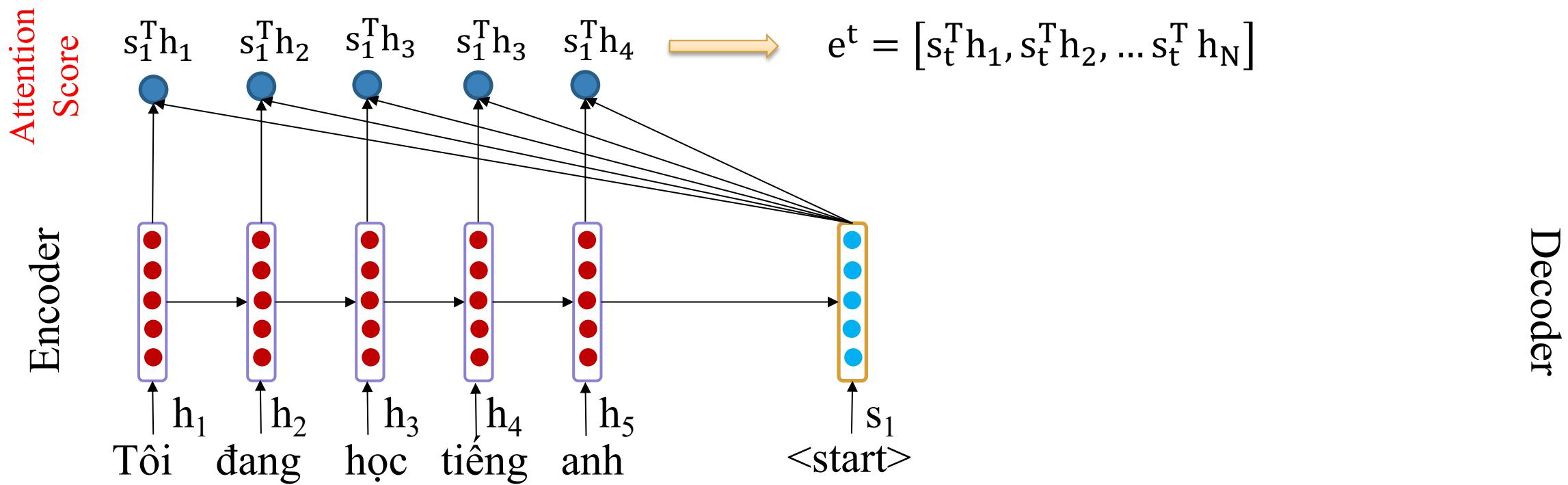
Seq2Seq with Attention



2 – Transformer

!

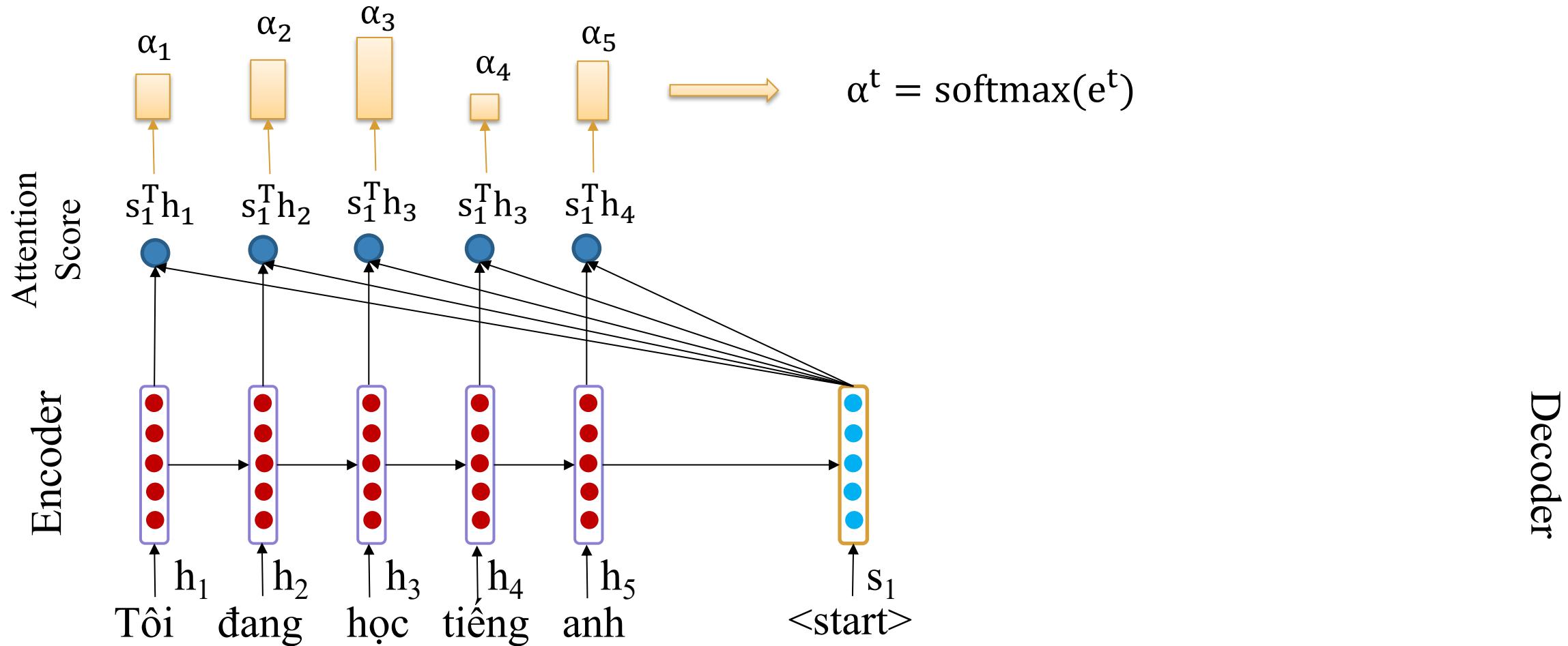
Seq2Seq with Attention

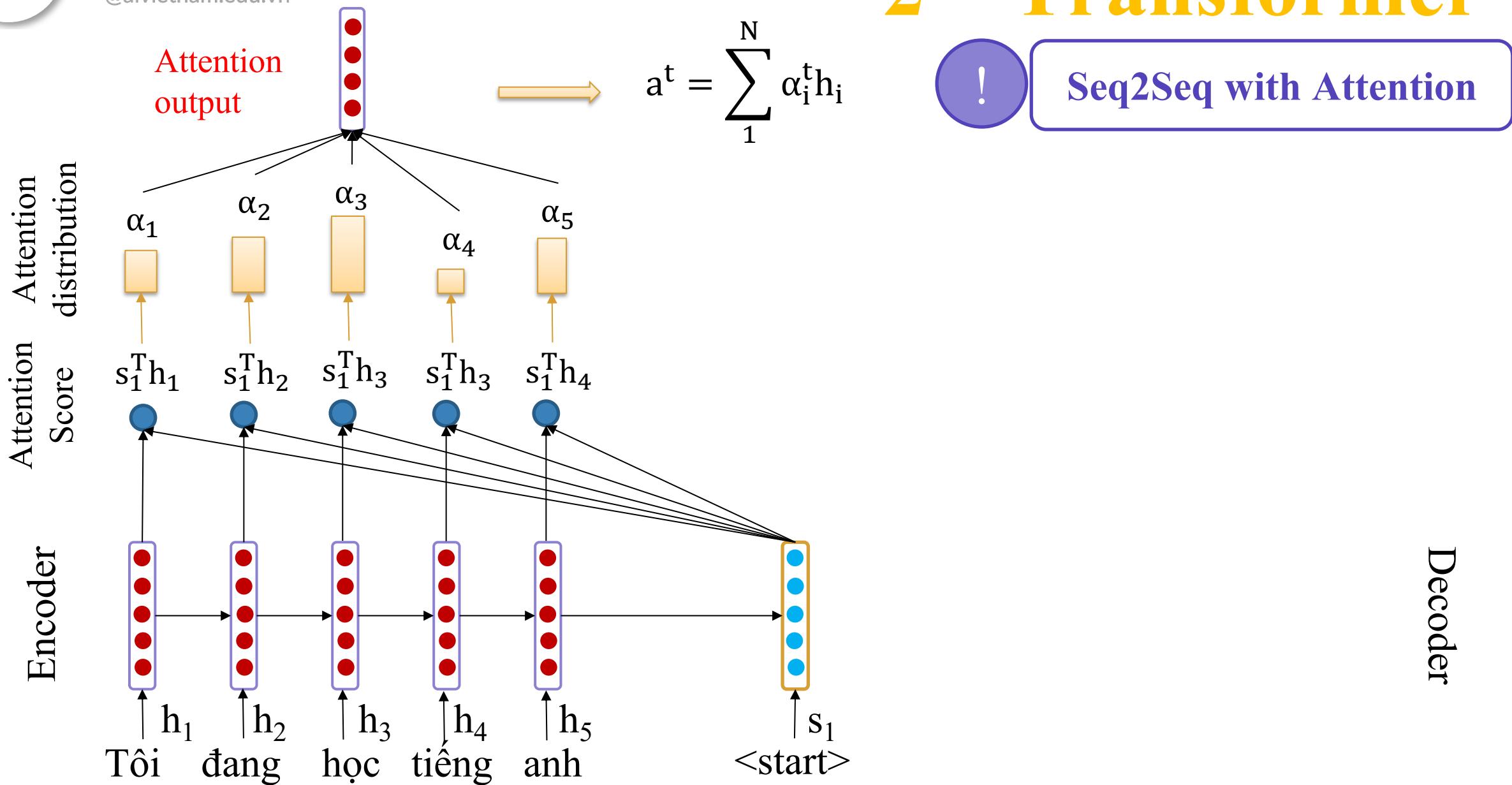


2 – Transformer

!

Seq2Seq with Attention



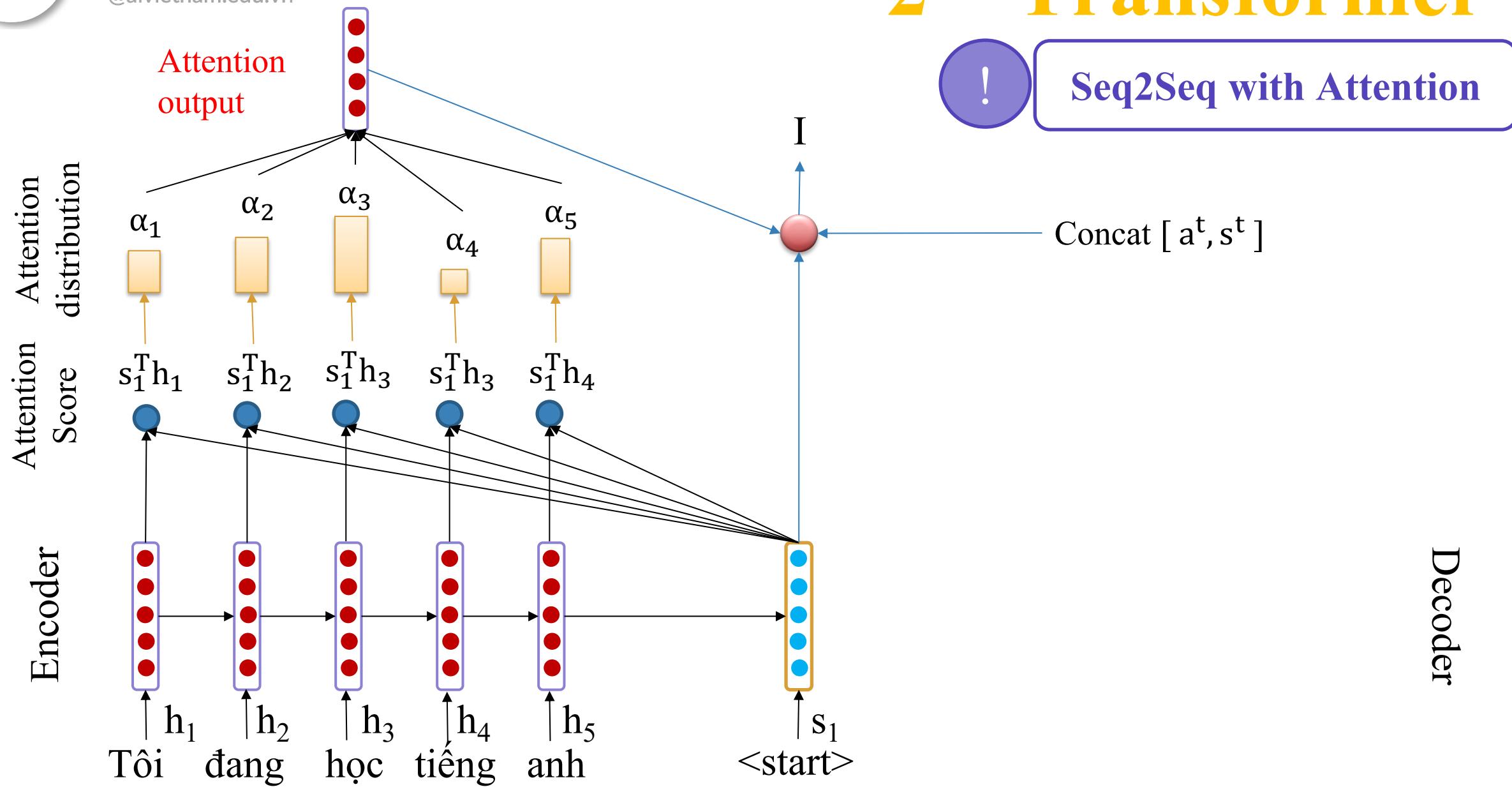


2 – Transformer

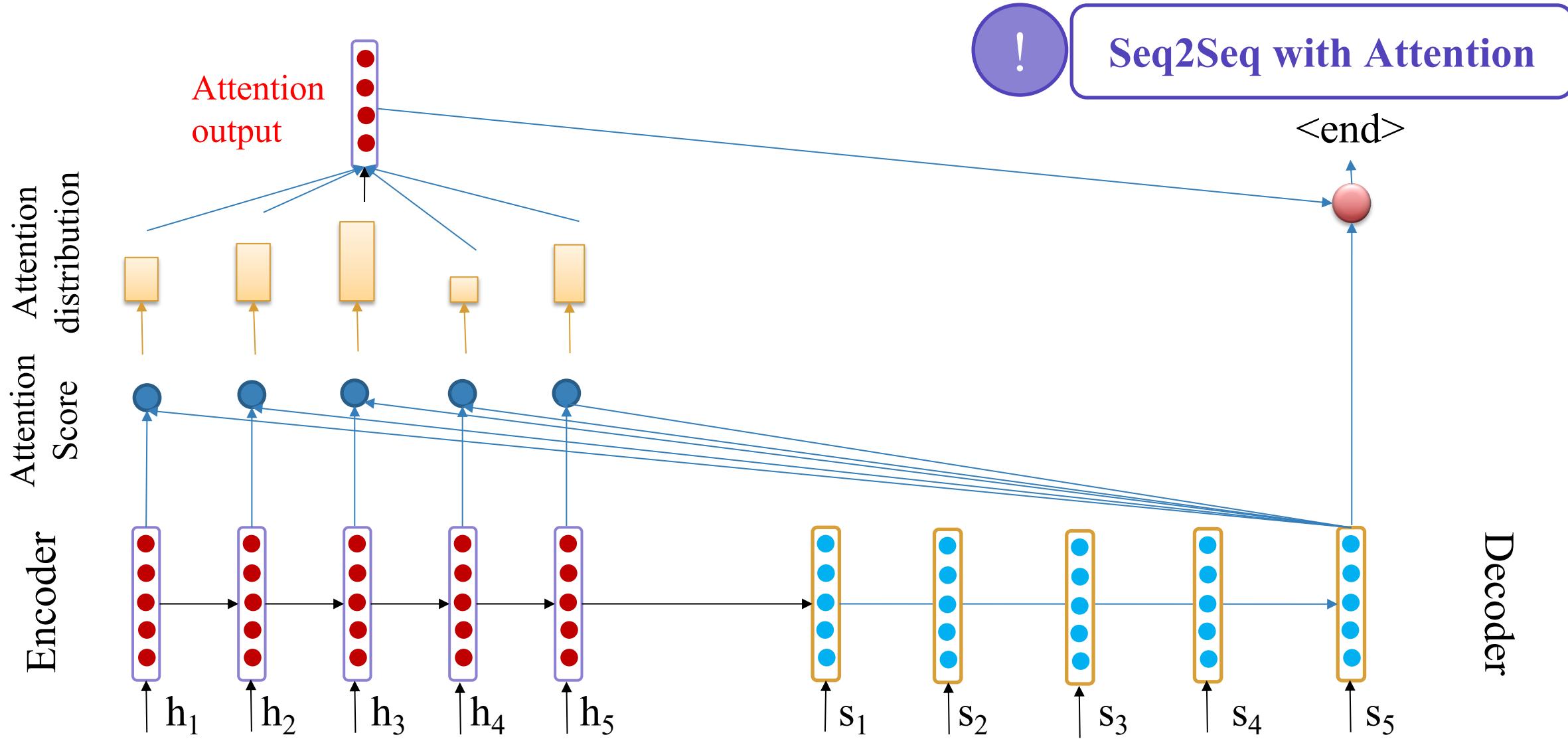


Seq2Seq with Attention

2 – Transformer



2 – Transformer



2 – Transformer



Seq2Seq with Attention

Compute $e \in \mathbb{R}^N$ from $h_1, h_2, \dots, h_N \in \mathbb{R}^{d_1}$ and $s \in \mathbb{R}^{d_2}$

- Dot-product attention

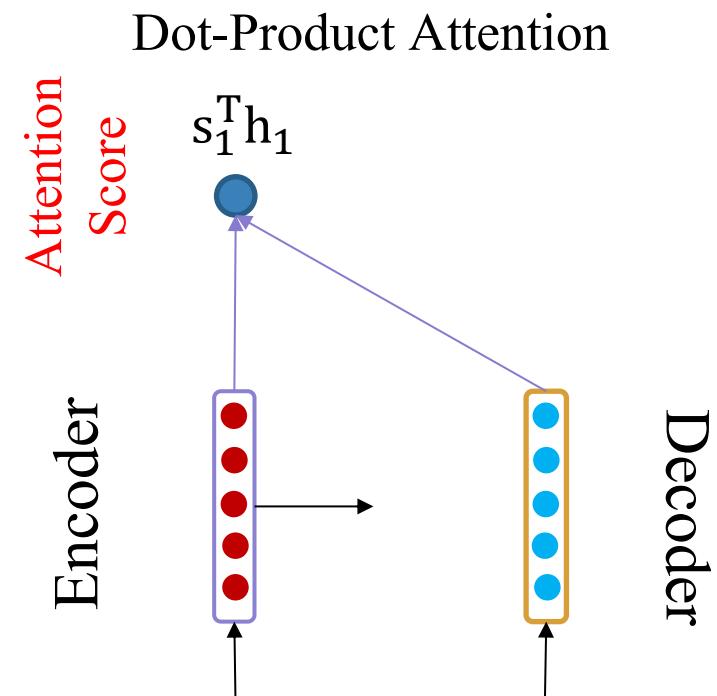
$$e_i = s^T h_i; d_1 = d_2$$

- Multiplicative attention

$$e_i = s^T W h_i; W \in \mathbb{R}^{d_2 \times d_1}$$

- Additive attention

$$\begin{aligned} e_i &= v^T \tanh(W_1 h_i + W_2 s) \\ W_1 &\in \mathbb{R}^{d_3 \times d_1}, W_2 \in \mathbb{R}^{d_3 \times d_2}, v \in \mathbb{R}^{d_3} \end{aligned}$$

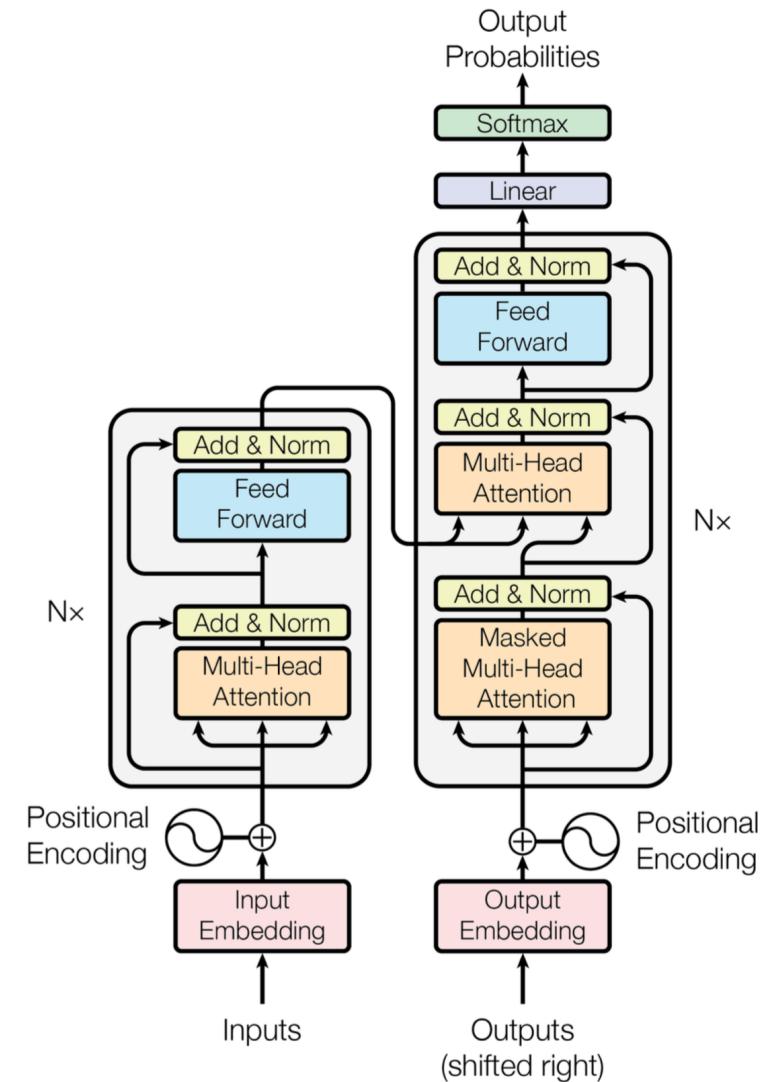


2 – Transformer



Transformer Architecture

- ❖ Architecture:
 - Token Embedding
 - Positional Embedding
 - N x Encoder Layer:
 - N x Decoder Layer
-
- ❖ Core technique: Self-Attention

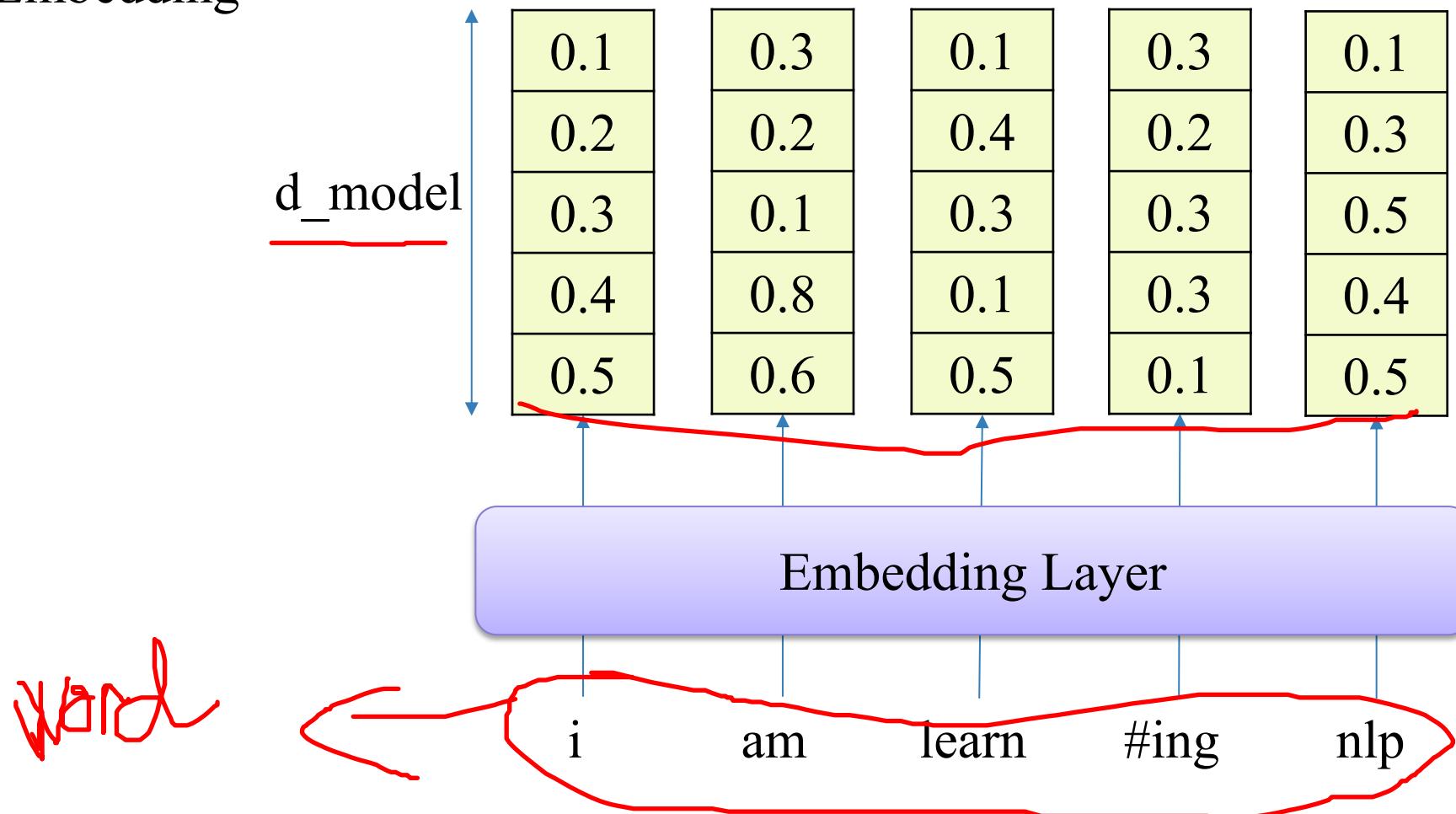


2 – Transformer



Transformer Architecture

➤ Token Embedding





Transformer Architecture

- Positional Embedding (Sinusoid)

$$\text{PE}_{(pos,2i)} = \sin\left(\frac{\text{pos}}{n^{\frac{d_{\text{model}}}{2}}}\right)$$
$$\text{PE}_{(pos,2i+1)} = \cos\left(\frac{\text{pos}}{n^{\frac{d_{\text{model}}}{2}}}\right)$$

pos: position of an entity in input sequence

d_{model} : dimension of the output embedding space

$n = 1000$ (recommend)

2 – Transformer



Transformer Architecture

- Positional Embedding (Sinusoid)

- Example: $n = 100$,
 $d_{model} = 5$

$$P_{00} = \sin\left(\frac{0}{100 \cdot \frac{5}{2}}\right)$$

$$P_{01} = \cos\left(\frac{0}{100 \cdot \frac{5}{2}}\right)$$

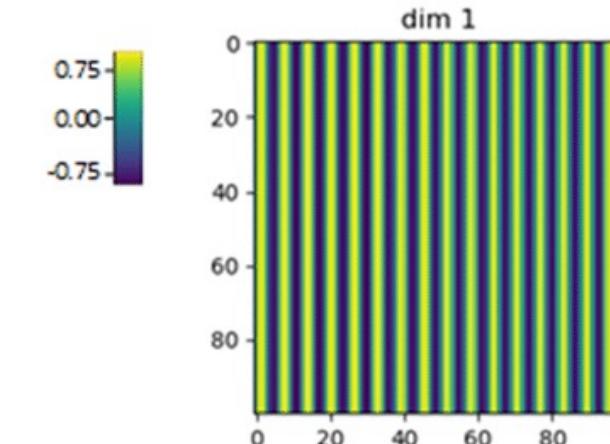
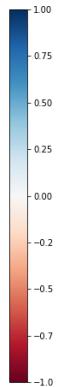
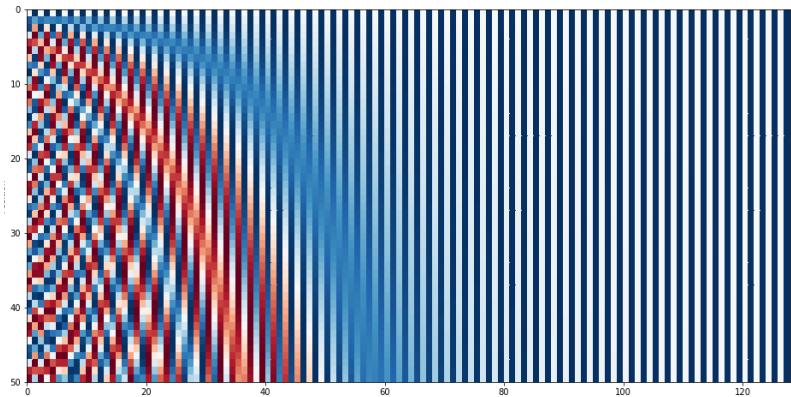
I	0	P ₀₀ = 0	P ₀₁ = 1	P ₀₂ = 0	P ₀₃ = 1	P ₀₄ = 0
am	1	P ₁₀ = 0.84	P ₁₁ =0.54	P ₁₂ =0.16	P ₁₃ =0.99	P ₁₄ =0.025
learn	2	P ₂₀ = 0.91	P ₂₁ =-0.42	P ₂₂ =0.31	P ₂₃ =0.95	P ₂₄ =0.05
#ing	3	P ₃₀ = 0.14	P ₃₁ =-0.99	P ₃₂ =0.46	P ₃₃ =0.89	P ₃₄ =0.141
nlp	4	P ₄₀ = -0.76	P ₄₁ =-0.65	P ₄₂ =0.59	P ₄₃ =0.81	P ₄₄ =0.1

2 – Transformer

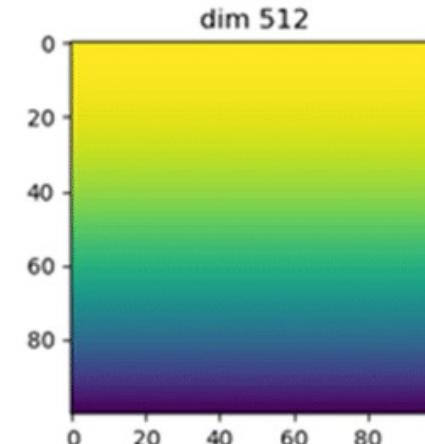


Transformer Architecture

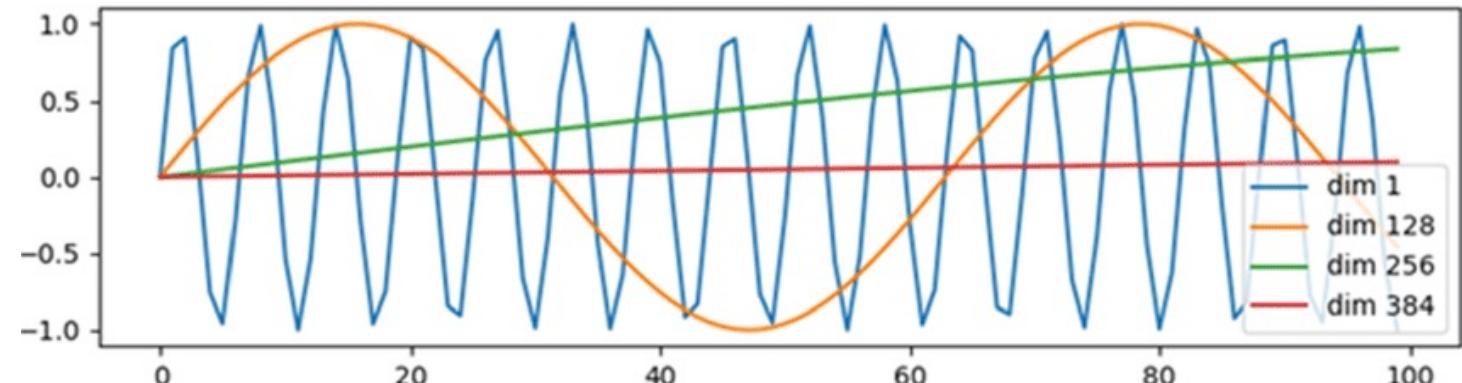
- Positional Embedding (Sinusoid)



dim 1



dim 512

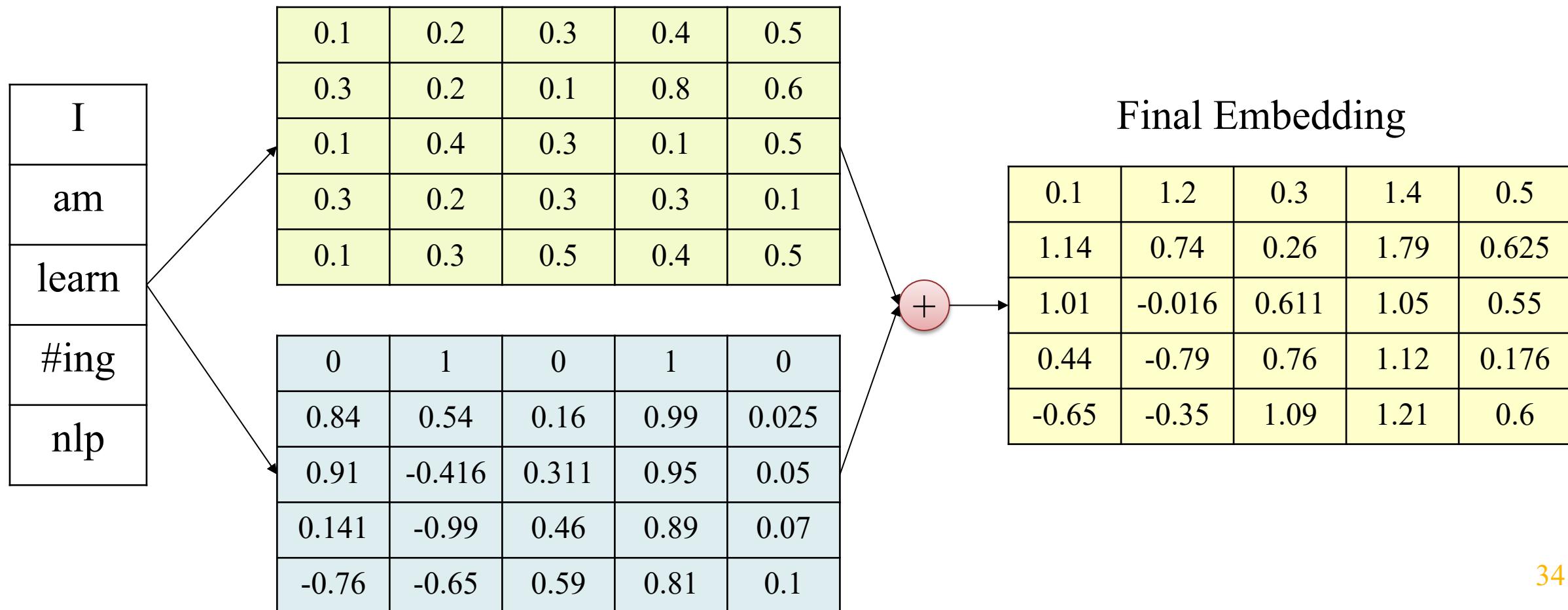


2 – Transformer

!

Transformer Architecture

- Token embedding with positional embedding

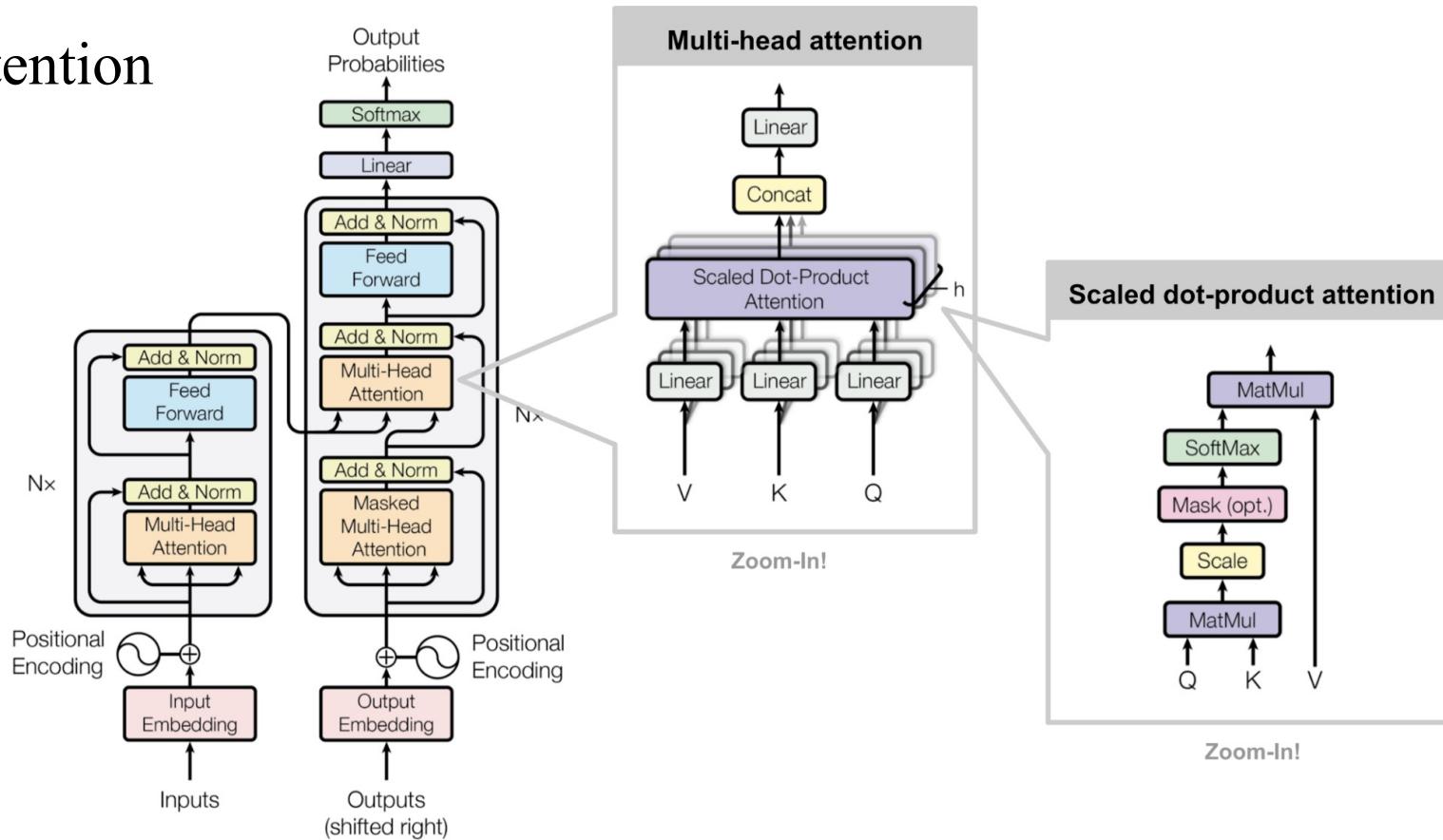


2 – Transformer



Transformer Architecture

- Transformer-Encoder
- Multi-Head Attention

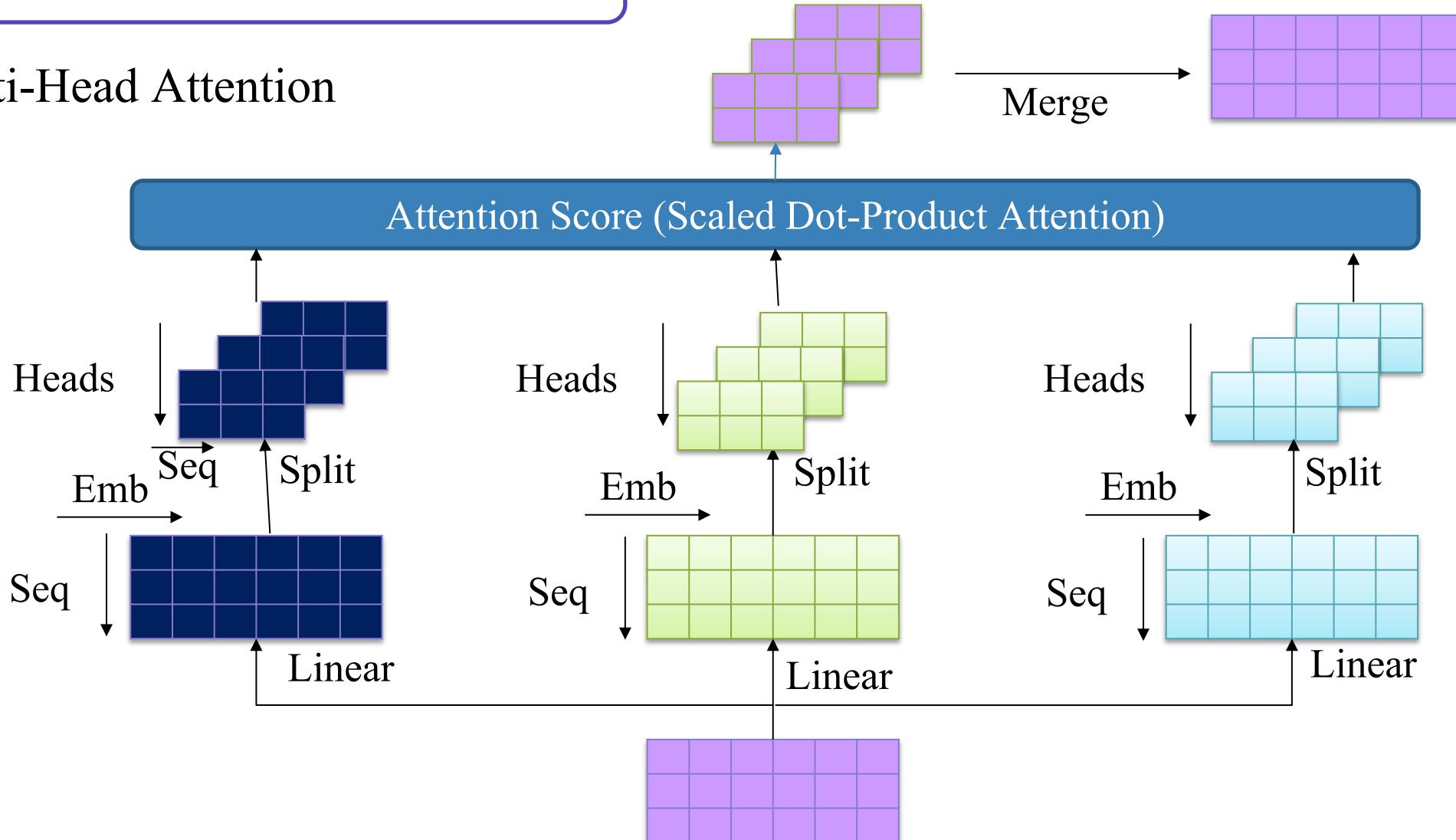


2 – Transformer



Transformer Architecture

- Multi-Head Attention



2 – Transformer



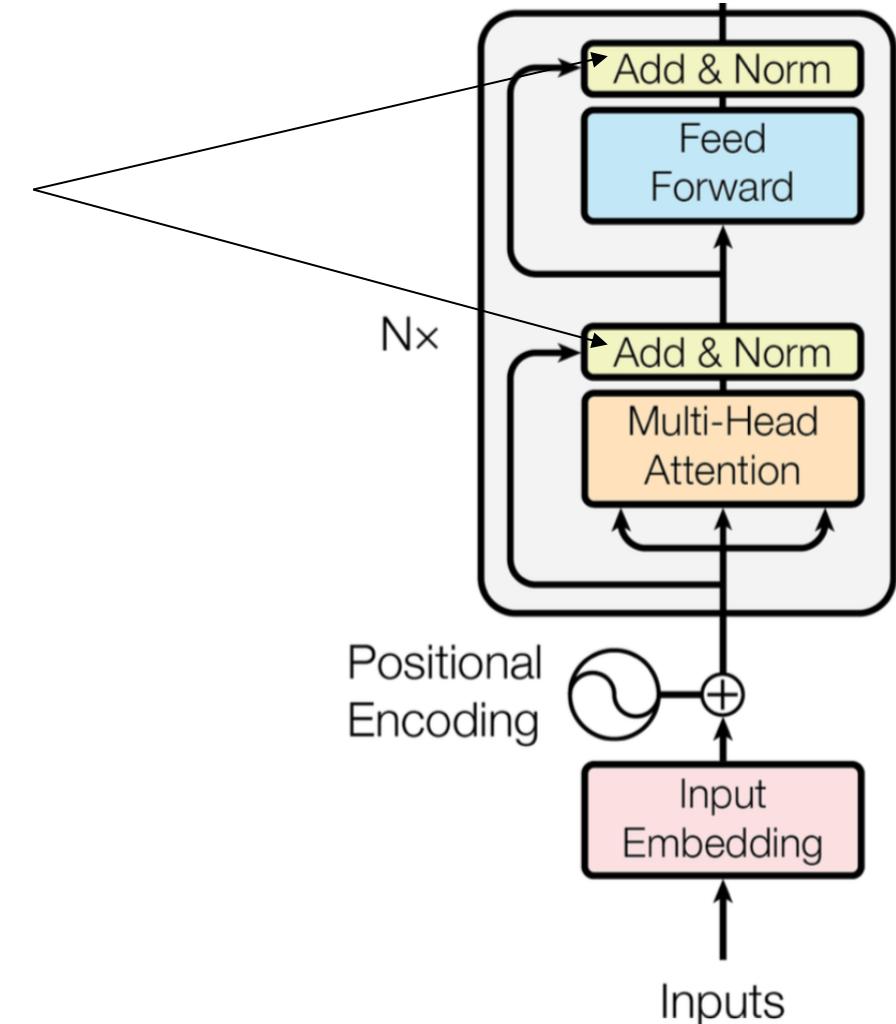
Transformer Architecture

- Transformer-Encoder
- Layer Normalization

$$\mu_i = \frac{1}{m} \sum_{j=1}^m x_{ij}$$

$$\sigma_i^2 = \frac{1}{m} \sum_{j=1}^m (x_{ij} - \mu_i)^2$$

$$\hat{x}_{ij} = \frac{(x_{ij} - \mu_i)}{\sqrt{\sigma_i^2 + \epsilon}}$$

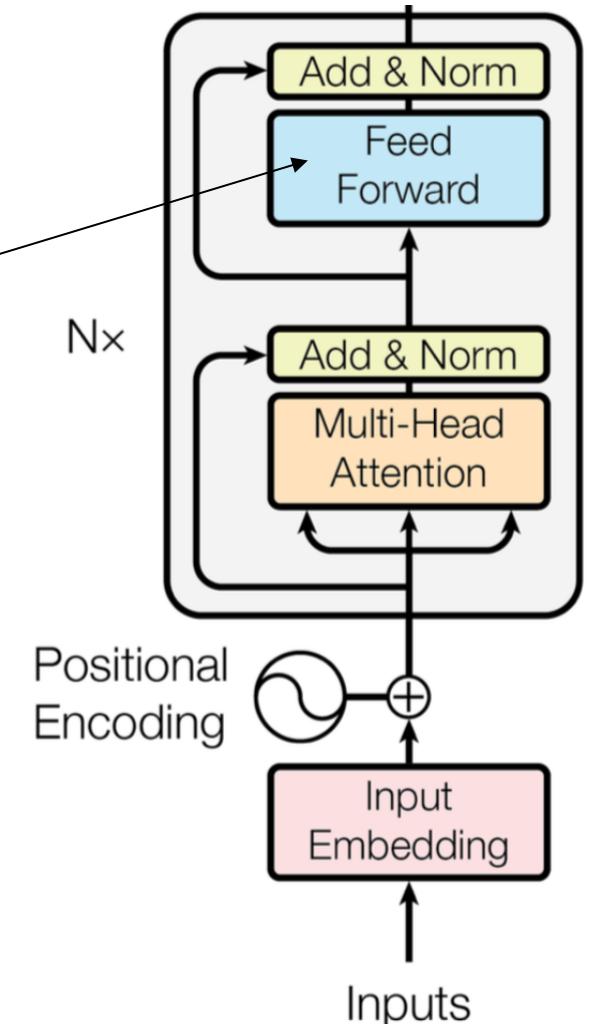
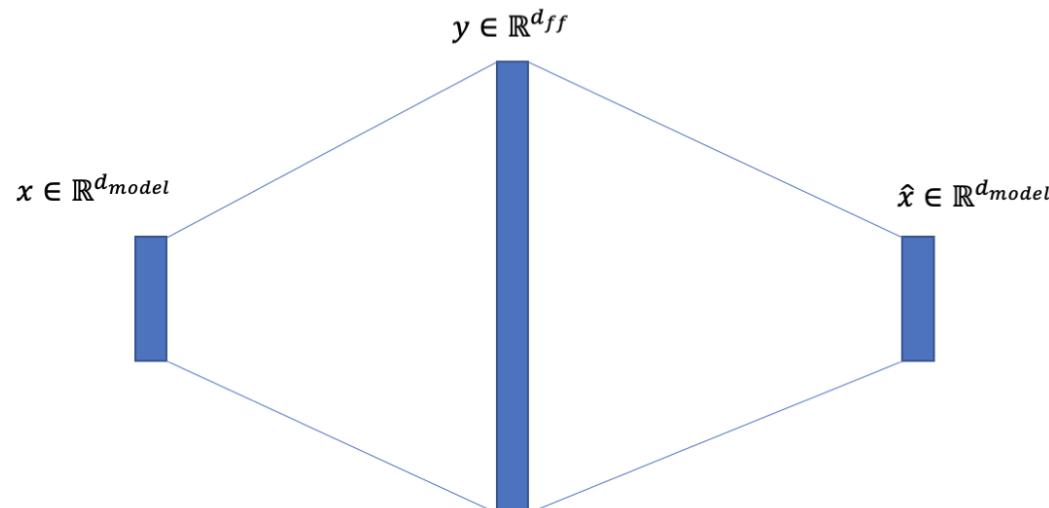


2 – Transformer



Transformer Architecture

- Transformer-Encoder
- Point Wise Feed Forward Network



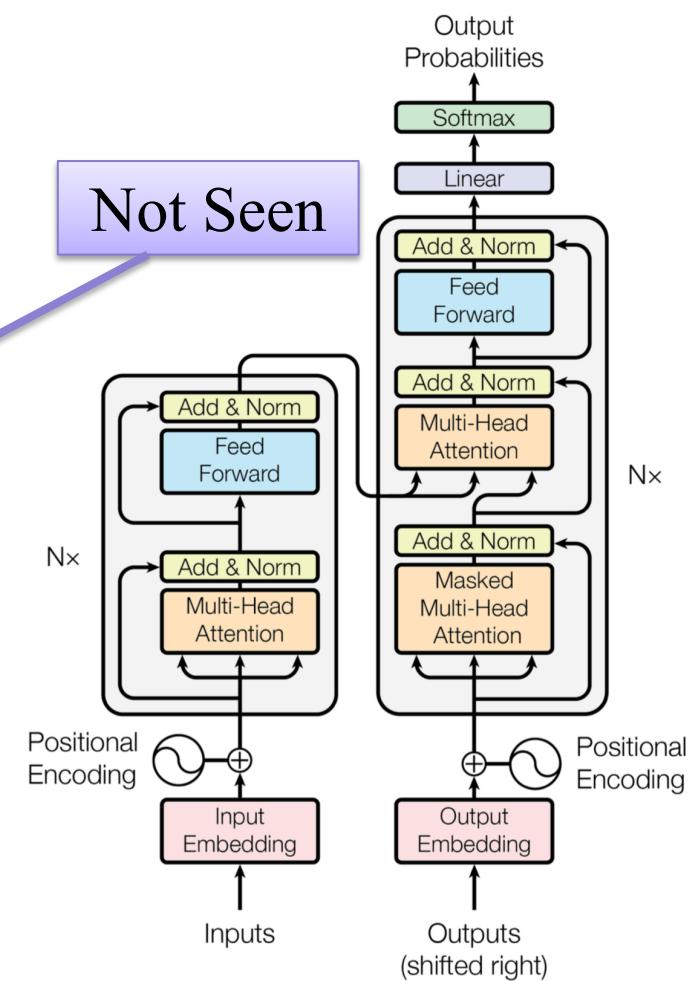
2 – Transformer



Transformer Architecture

- Transformer-Decoder
- Masked Multi-Head Attention
- Predict next token based on the history token

	I	am	learn	#ing	nlp
I	1.2	2.3	2.5	1.2	3.5
am	2.3	1.5	0.5	0.6	1.2
learn	0.5	1.4	1.6	0.3	4.8
#ing	0.6	1.8	2.4	0.3	1.2
nlp	2.1	2.3	0.2	2.0	2.5



2 – Transformer



Transformer Architecture

- Transformer-Decoder
- Masked Multi-Head Attention
- Predict next token based on the history to

I
am
learn
#ing
nlp

	I	am	learn	#ing	nlp
I	1.2	2.3	2.5	1.2	3.5
am	2.3	1.5	0.5	0.6	1.2
learn	0.5	1.4	1.6	0.3	4.8
#ing	0.6	1.8	2.4	0.3	1.2
nlp	2.1	2.3	0.2	2.0	2.5

1.2	-inf	-inf	-inf	-inf
2.3	1.5	-inf	-inf	-inf
0.5	1.4	1.6	-inf	-inf
0.6	1.8	2.4	0.3	-inf
2.1	2.3	0.2	2.0	2.5

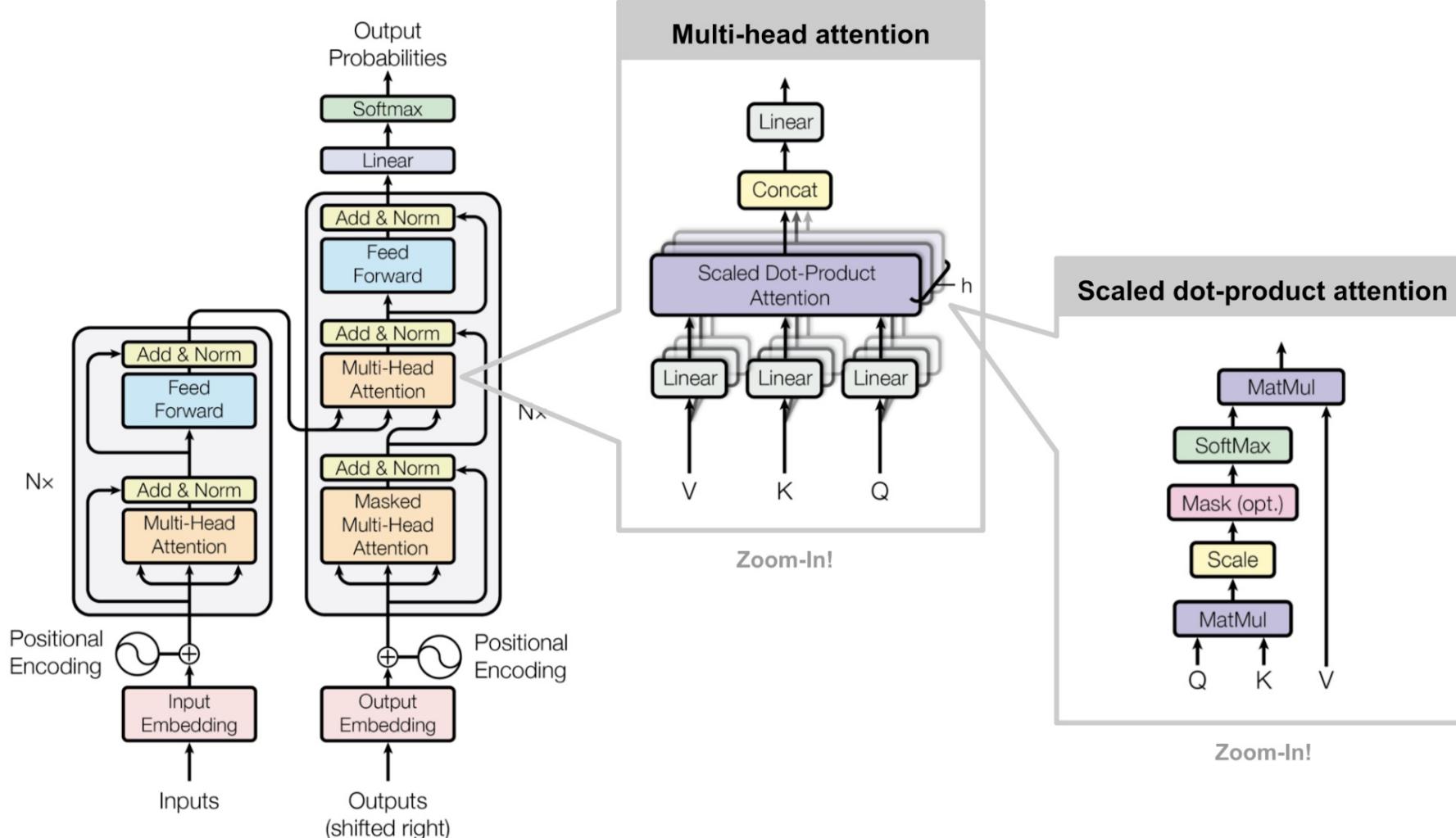
Softmax

1.0	0.0	0.0	0.0	0.0
0.69	0.31	0.0	0.0	0.0
0.15	0.38	0.46	0.0	0.0
0.6	1.8	2.4	0.3	0.0
2.1	2.3	0.2	2.0	2.5

2 – Transformer



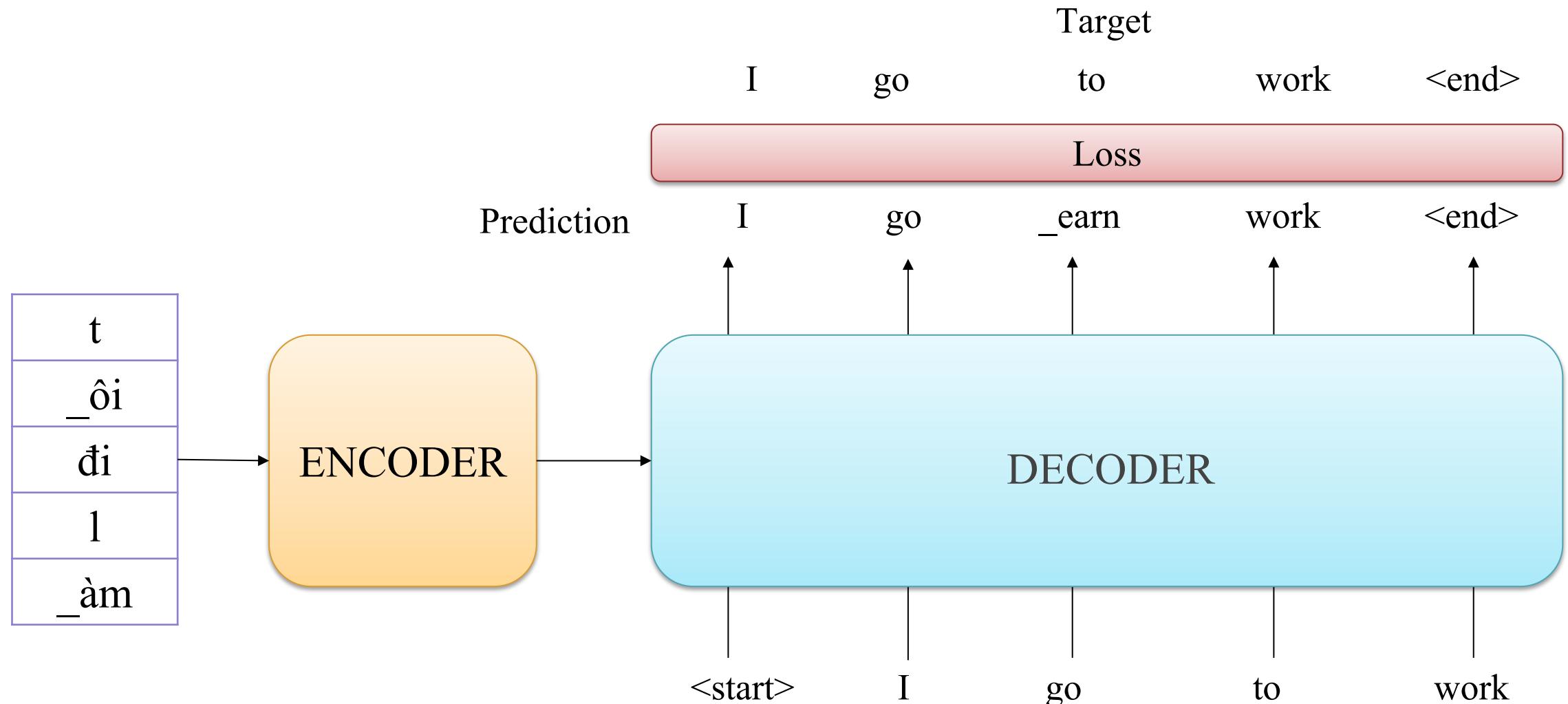
Transformer Architecture



2 – Transformer

!

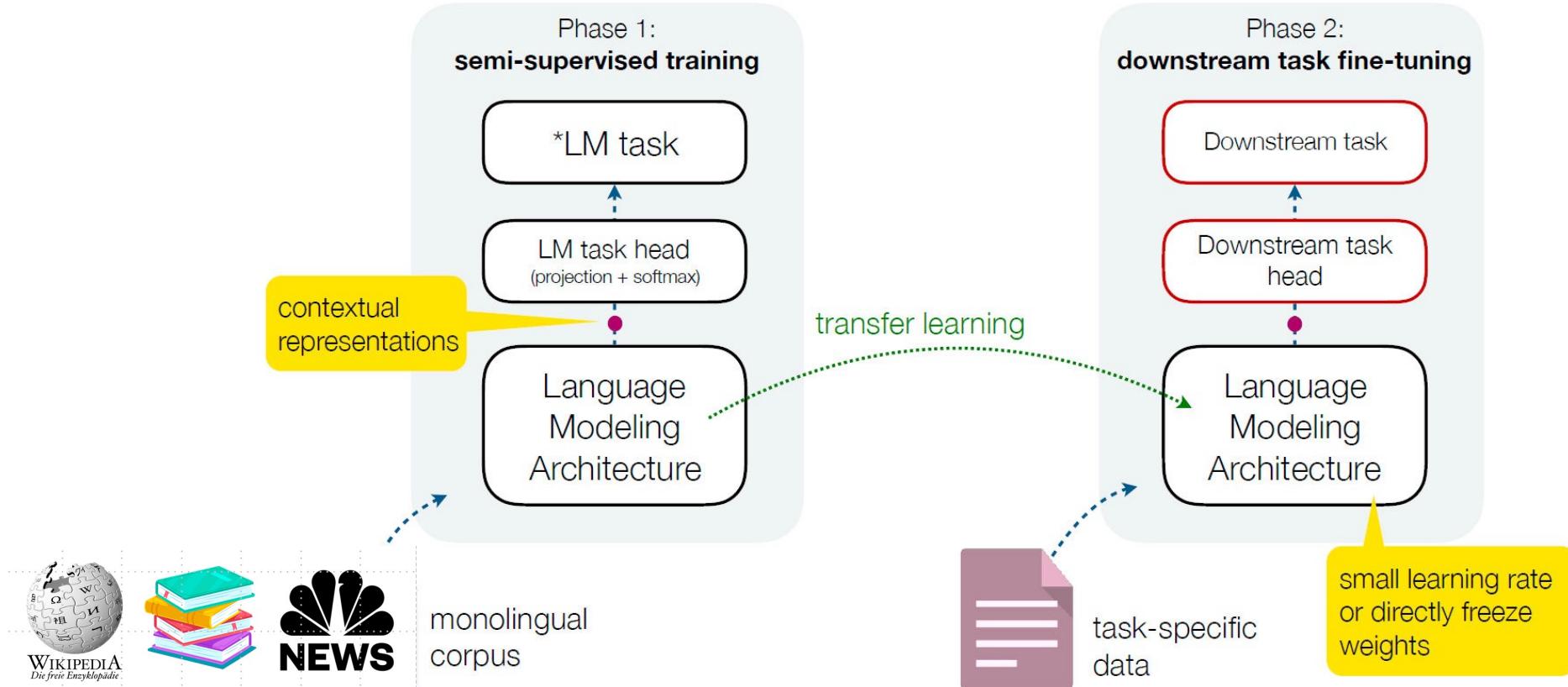
Transformer Architecture



3 – Pre-trained LMs



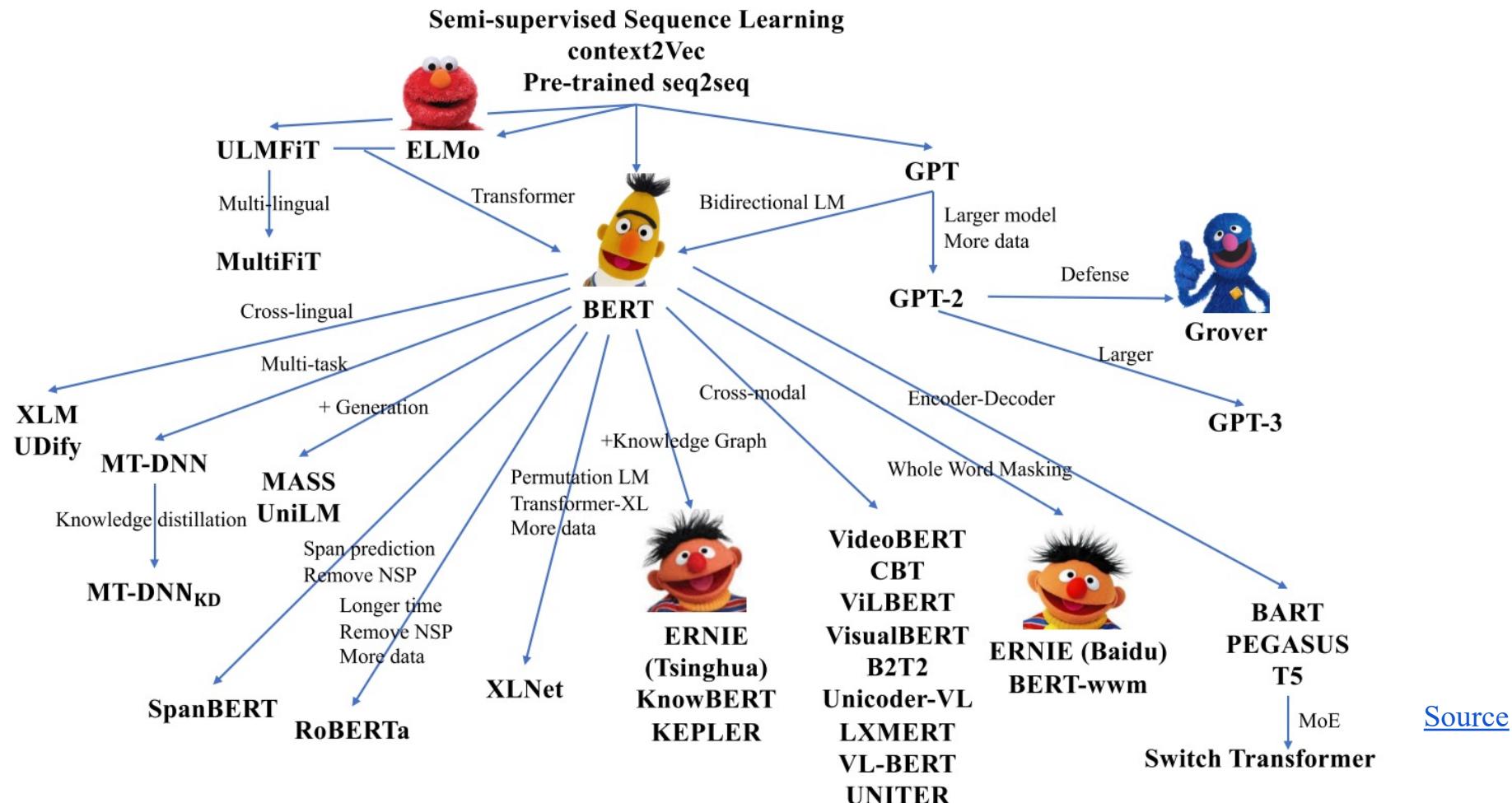
Transfer Learning



3 – Pre-trained LMs

!

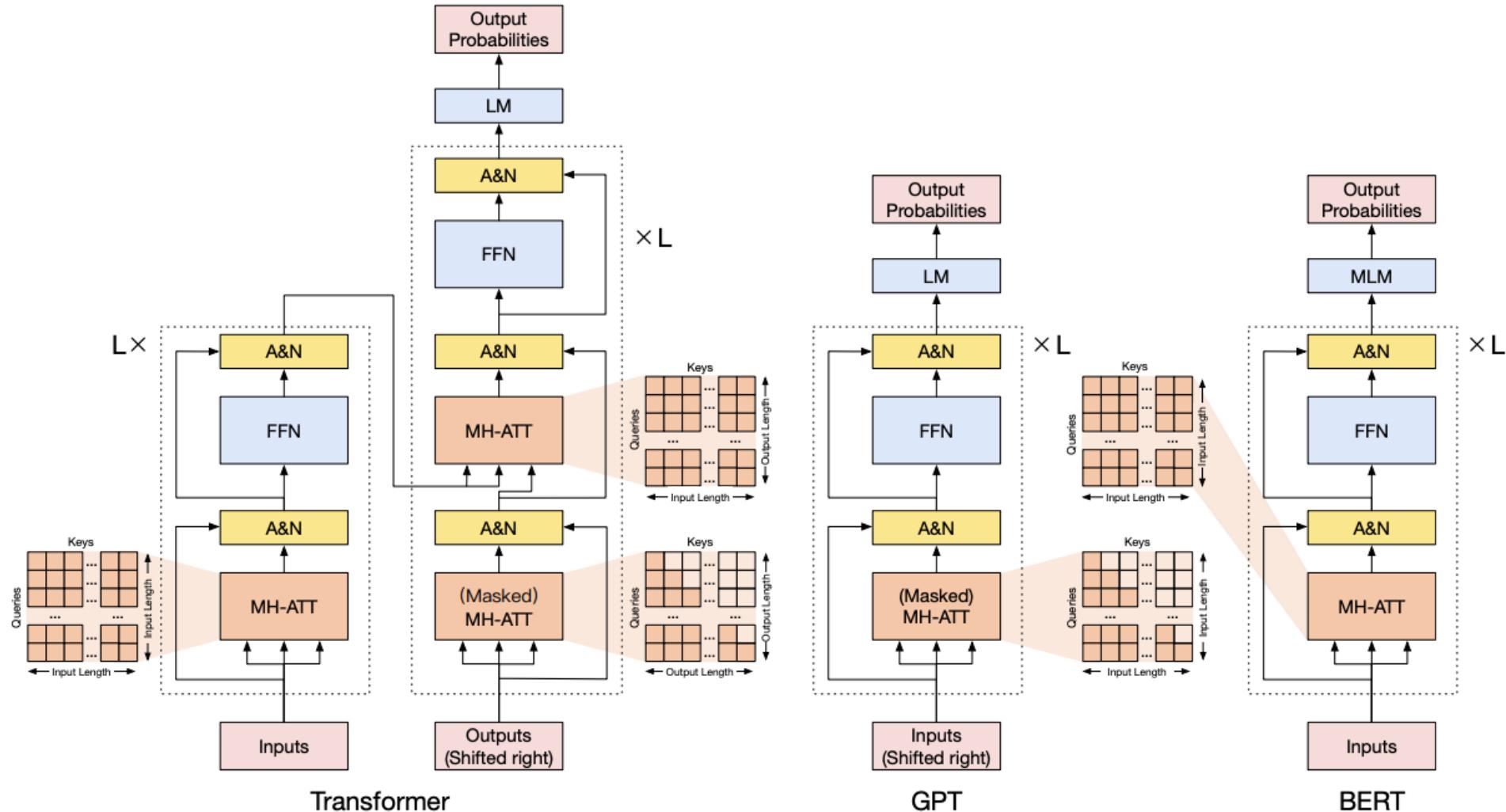
The family of recent typical pre-trained language models (PTMs)



3 – Pre-trained LMs



The architecture of Transformer, GPT and BERT

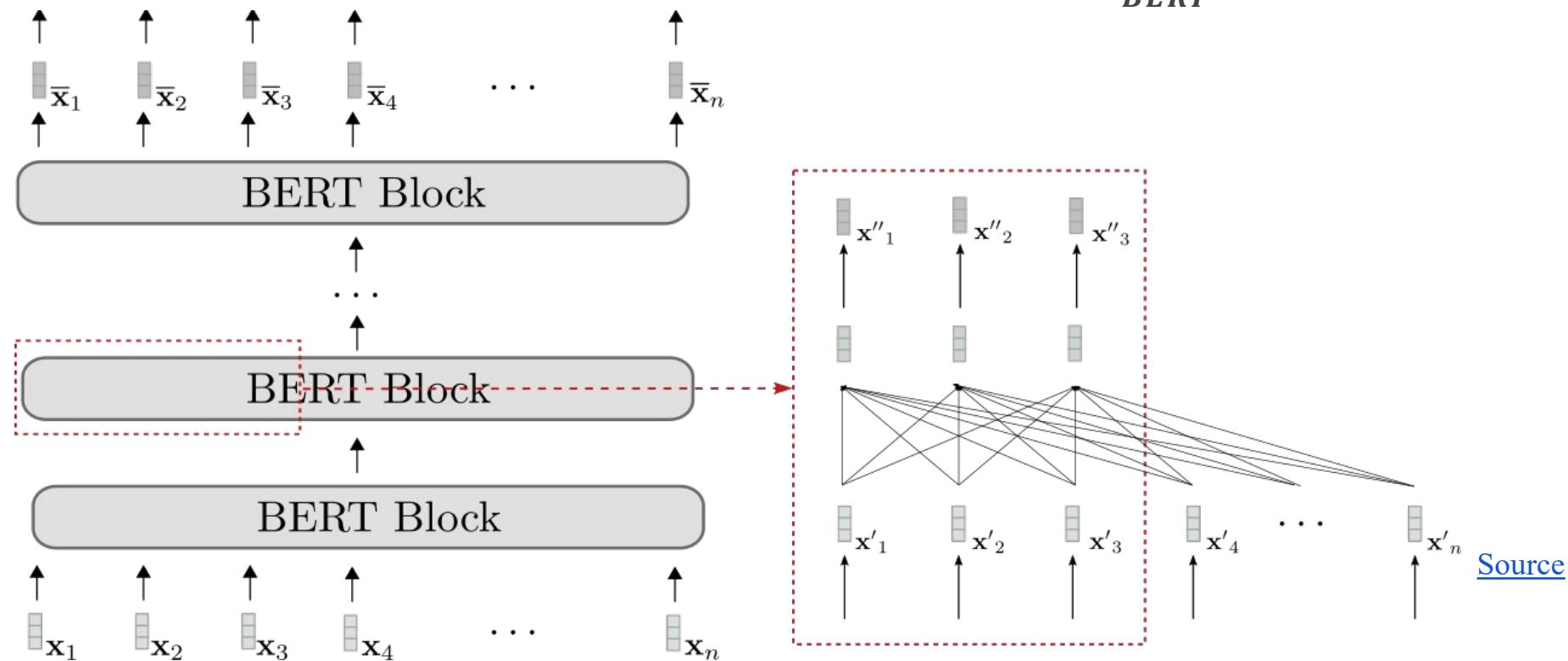


3 – Pre-trained LMs



BERT

- ❖ BERT: An encoder-only model
- ❖ Maps an input sequence to a contextualized sequence: $f_{\theta_{BERT}}: X_{1:n} \rightarrow \bar{X}_{1:n}$



3 – Pre-trained LMs



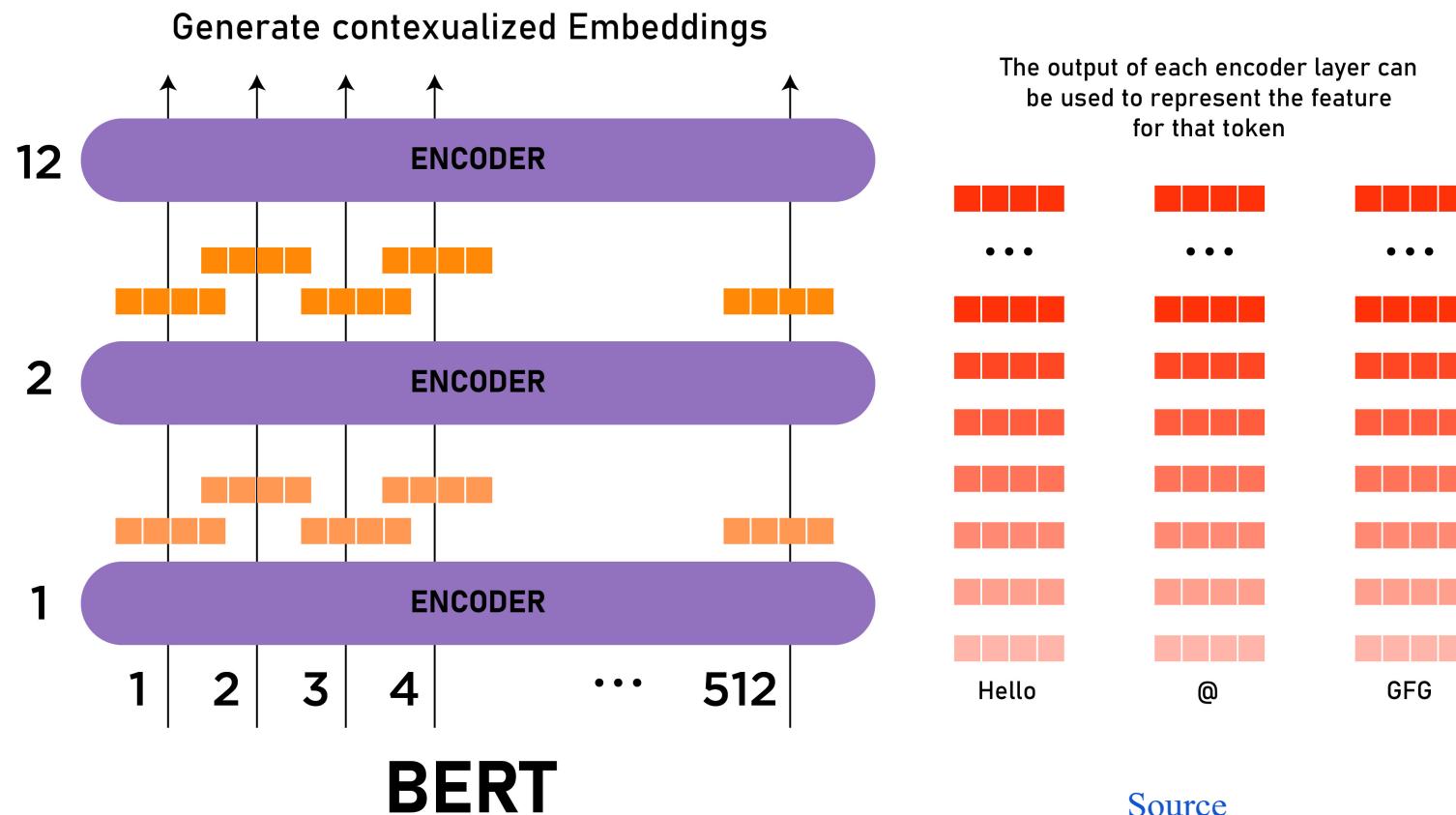
BERT

❖ BERT base:

L=12, H=768, A=12, P=110M

❖ BERT large:

L=24, H=1024, A=16, P=340M



[Source](#)

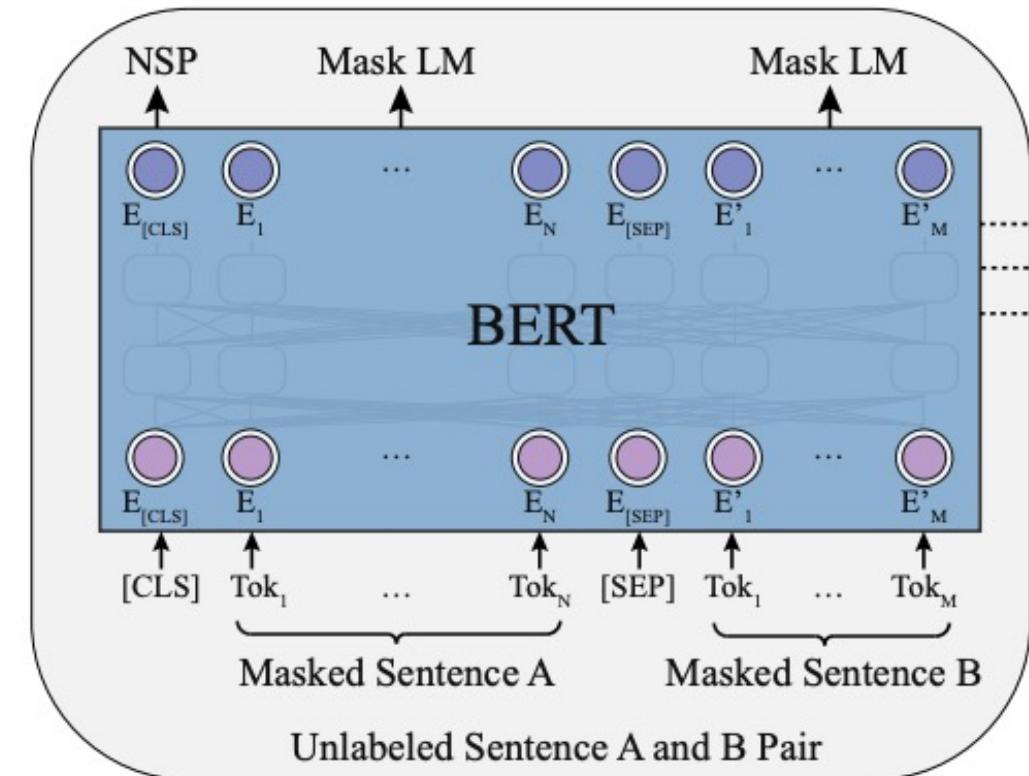
3 – Pre-trained LMs



BERT

Objective Functions:

- ❖ Masked LM (15% token):
 - 80%: replace with [MASK]
 - 10%: replace with a random word
 - 10%: keep unchanged
- ❖ Next Sentence Prediction (NSP)
 - Classification Task
 - 2 Labels: IsNext and NotNext
 - Use [SEP] token



Pre-training

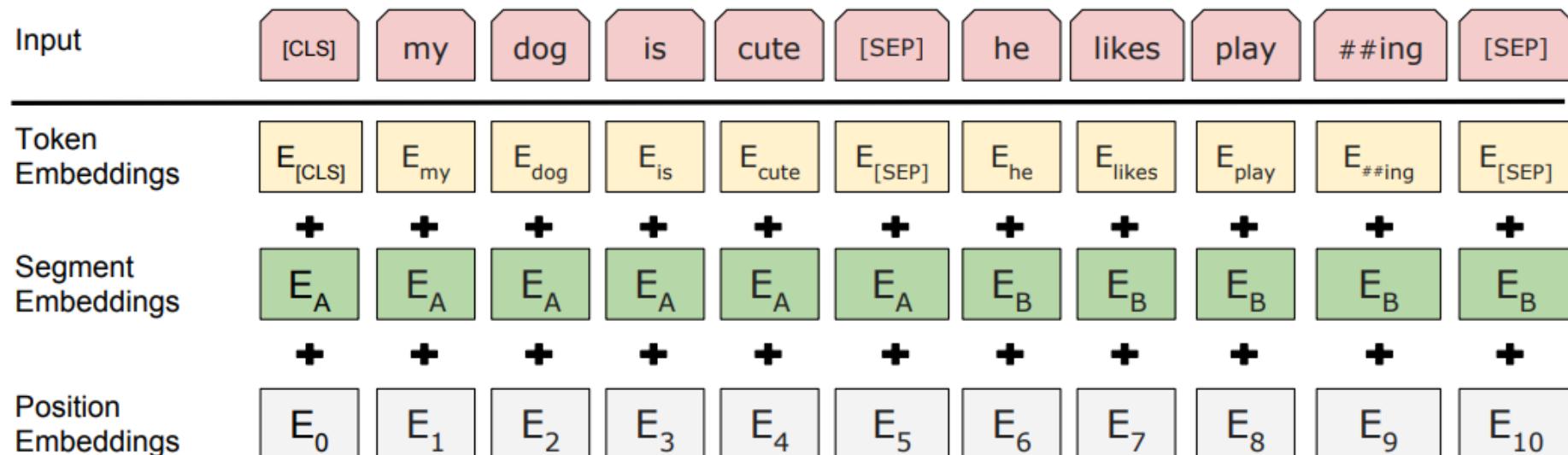
[Source](#)

3 – Pre-trained LMs

!

BERT

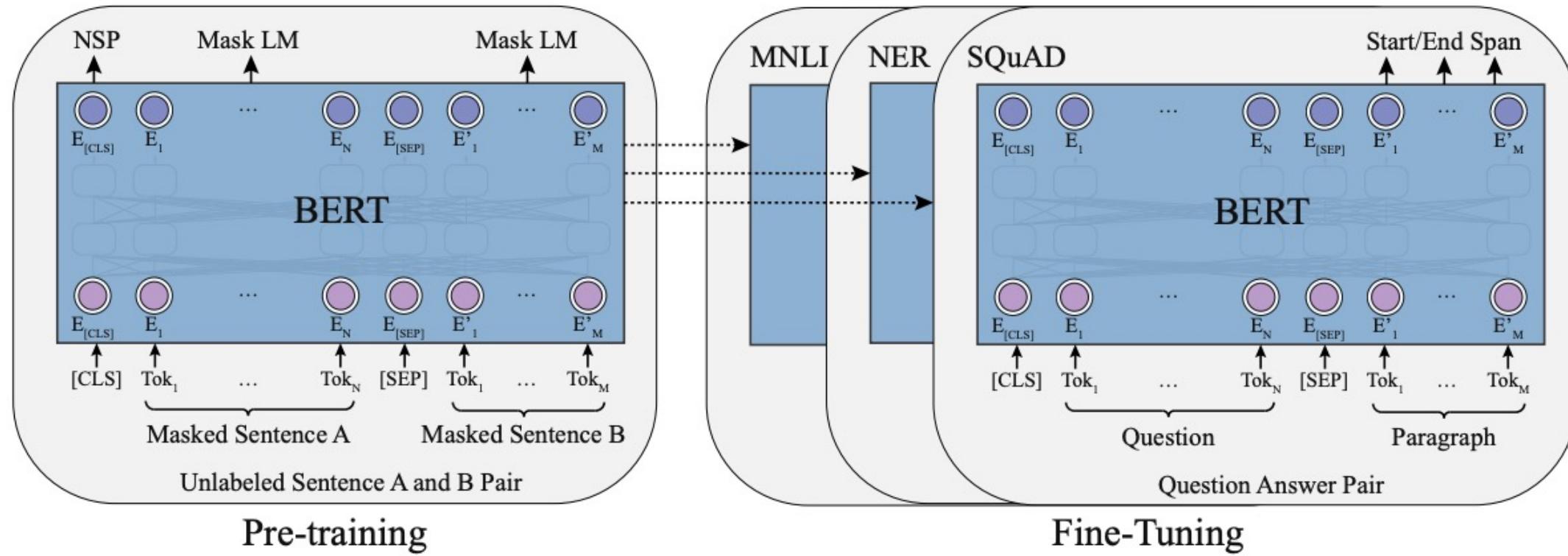
❖ BERT input representation

[Source](#)

3 – Pre-trained LMs



BERT



[Source](#)

3 – Pre-trained LMs



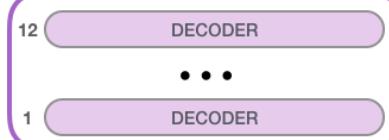
GPT (GPT2)

- ❖ GPT2: A decoder-only model, use uni-directional (causal) self-attention
- ❖ Maps an input sequence to a “next word” logit vector sequence:

$$f_{\theta_{GPT2}}: X_{0:m-1} \rightarrow L_{1:m}$$



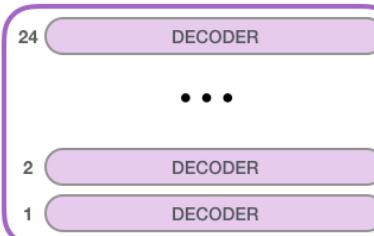
GPT-2
SMALL



Model Dimensionality: 768



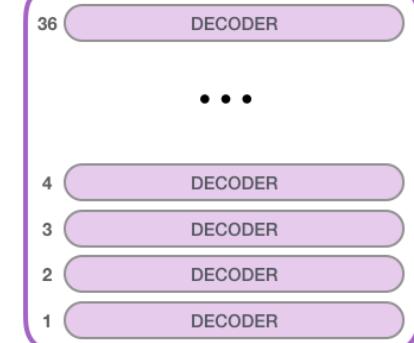
GPT-2
MEDIUM



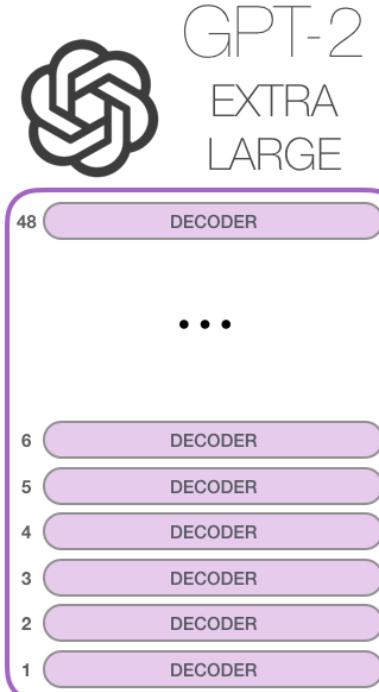
Model Dimensionality: 1024



GPT-2
LARGE



Model Dimensionality: 1280



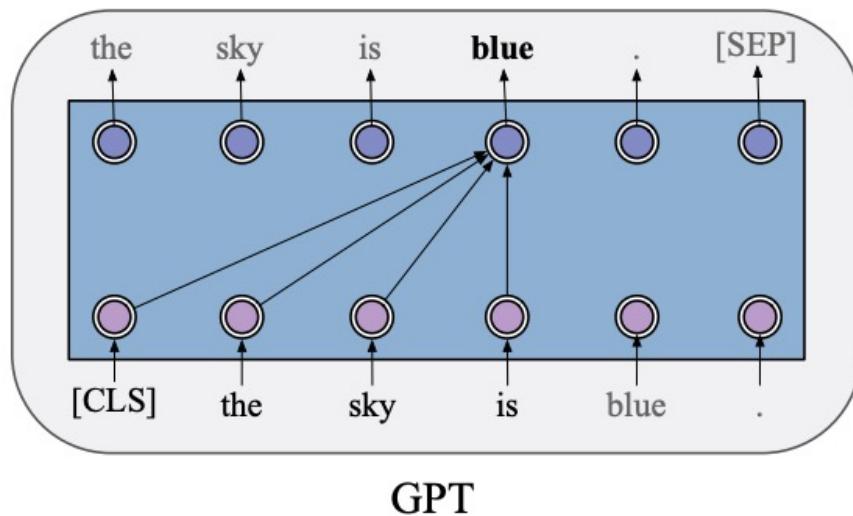
Model Dimensionality: 1600

GPT-2
EXTRA
LARGE

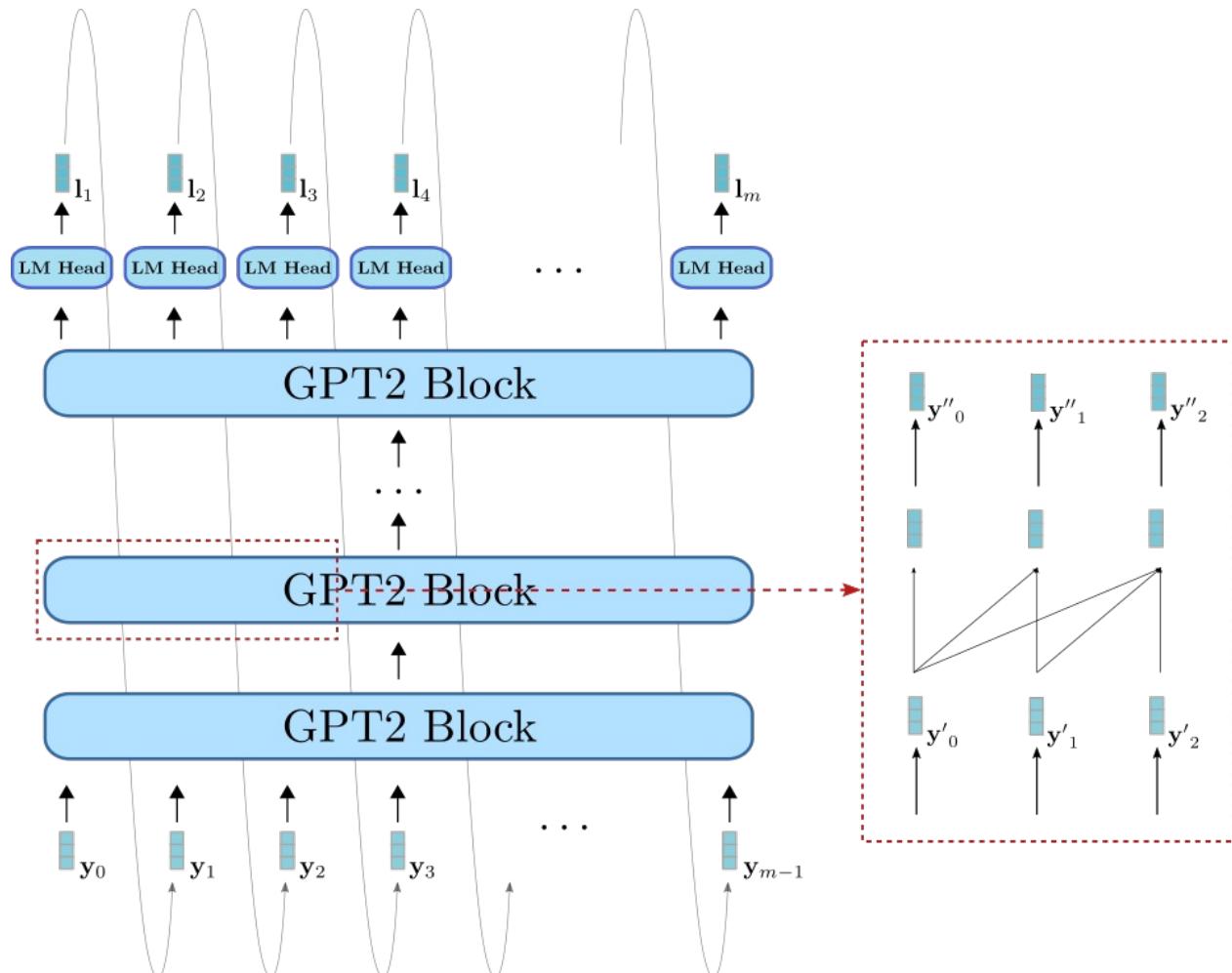
3 – Pre-trained LMs



GPT (GPT2)



GPT

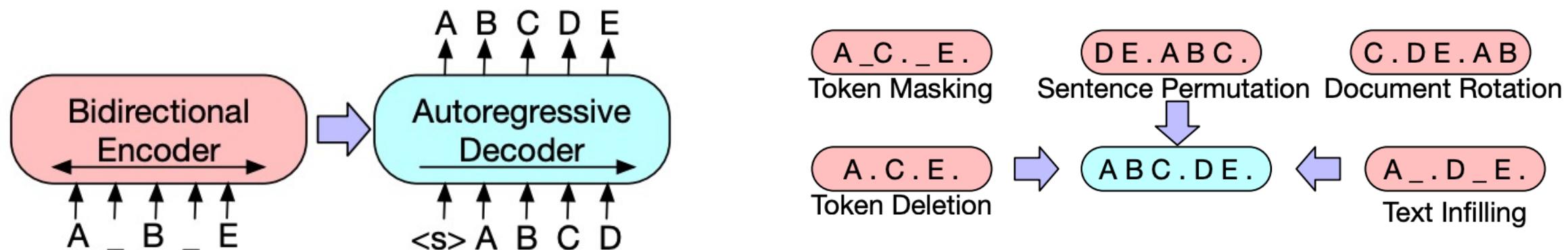


3 – Pre-trained LMs



BART

- ❖ **BART:** Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation and Comprehension

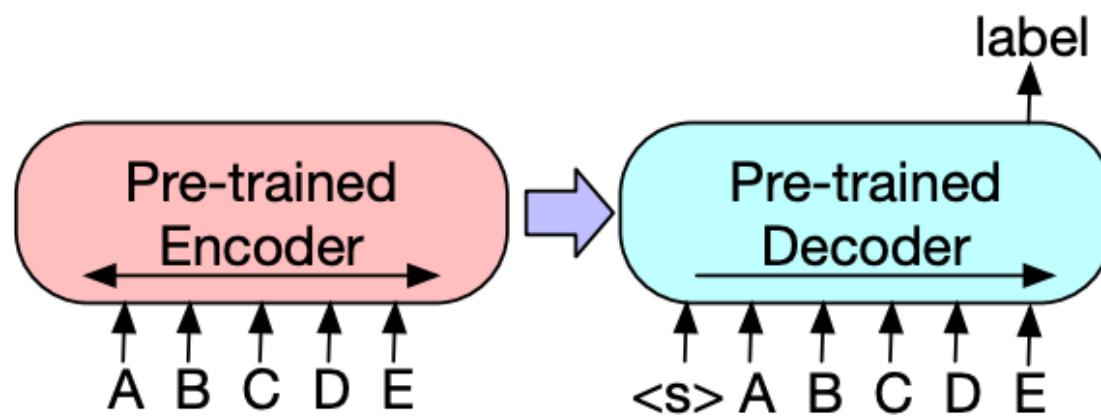


3 – Pre-trained LMs

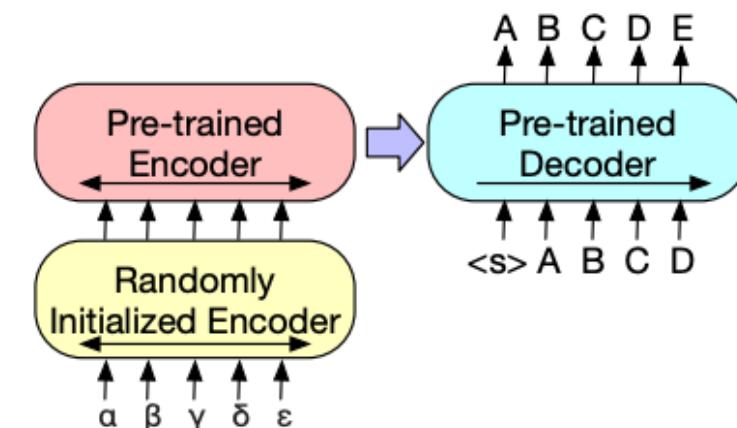


BART

- ❖ **BART**: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation and Comprehension
- ❖ Fine-tune BART



Classification Task



Machine Translation Task

4 – Applications

!

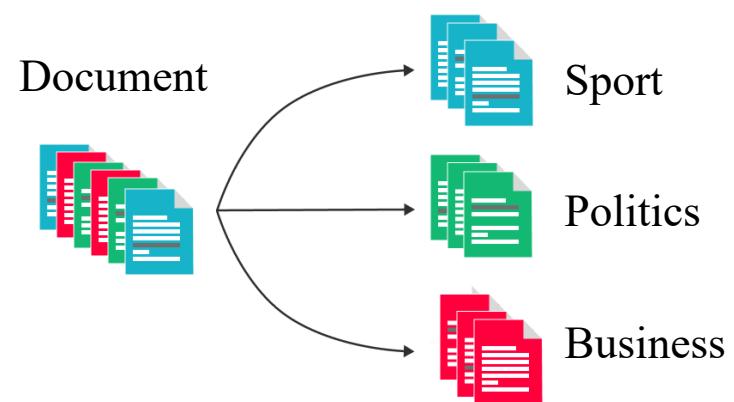
Text Classification

Input

- A document d
- A fixed set of classes $C = \{c_1, c_2, \dots, c_j\}$
- A training set of m hand-labeled documents: $(d_1, c_1), \dots, (d_m, c_j)$

Output

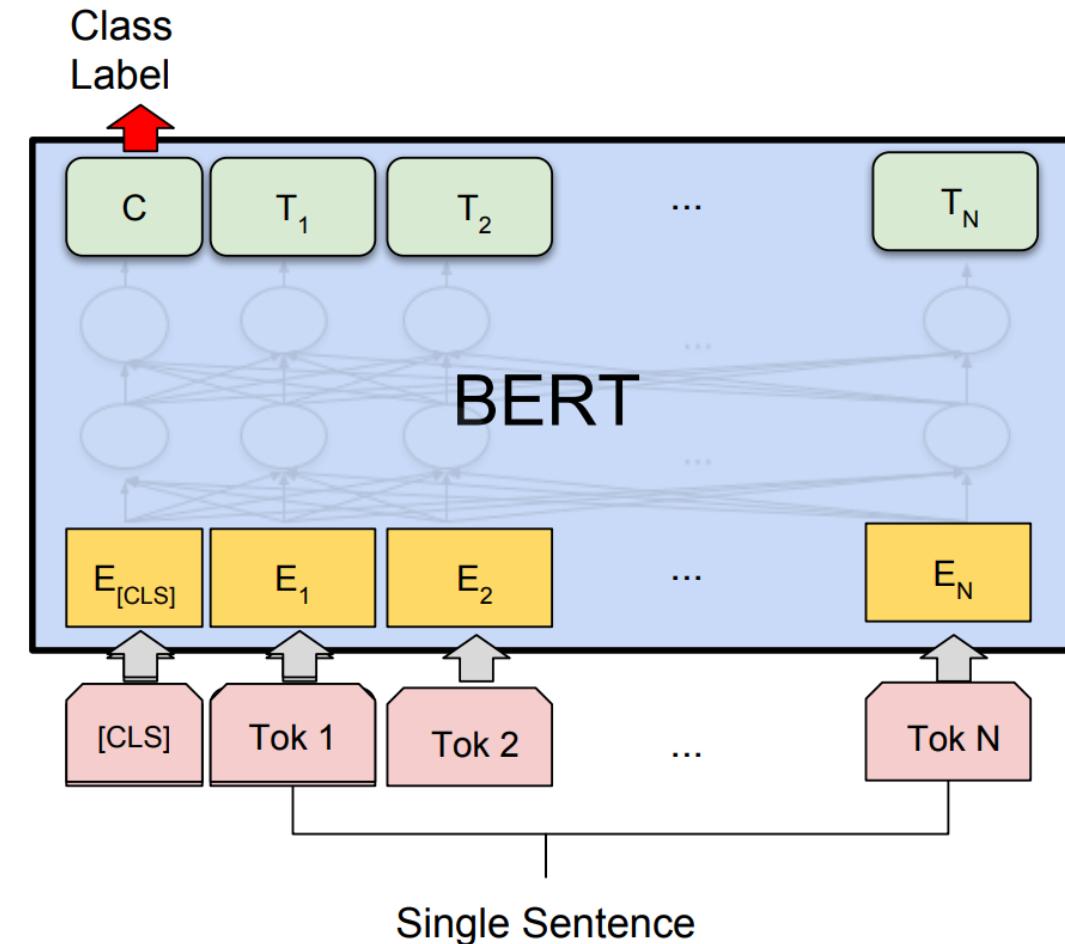
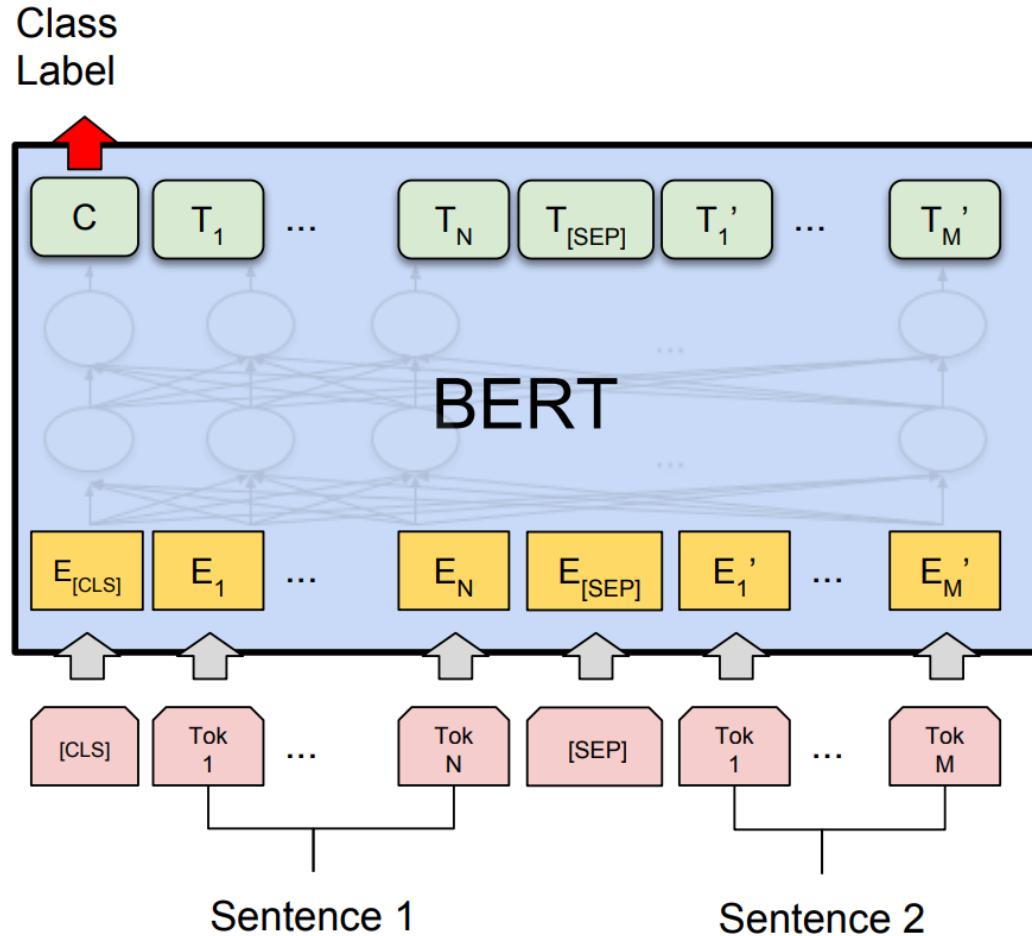
- A learned classifier $d \Rightarrow c$



4 – Applications



Text Classification using BERT

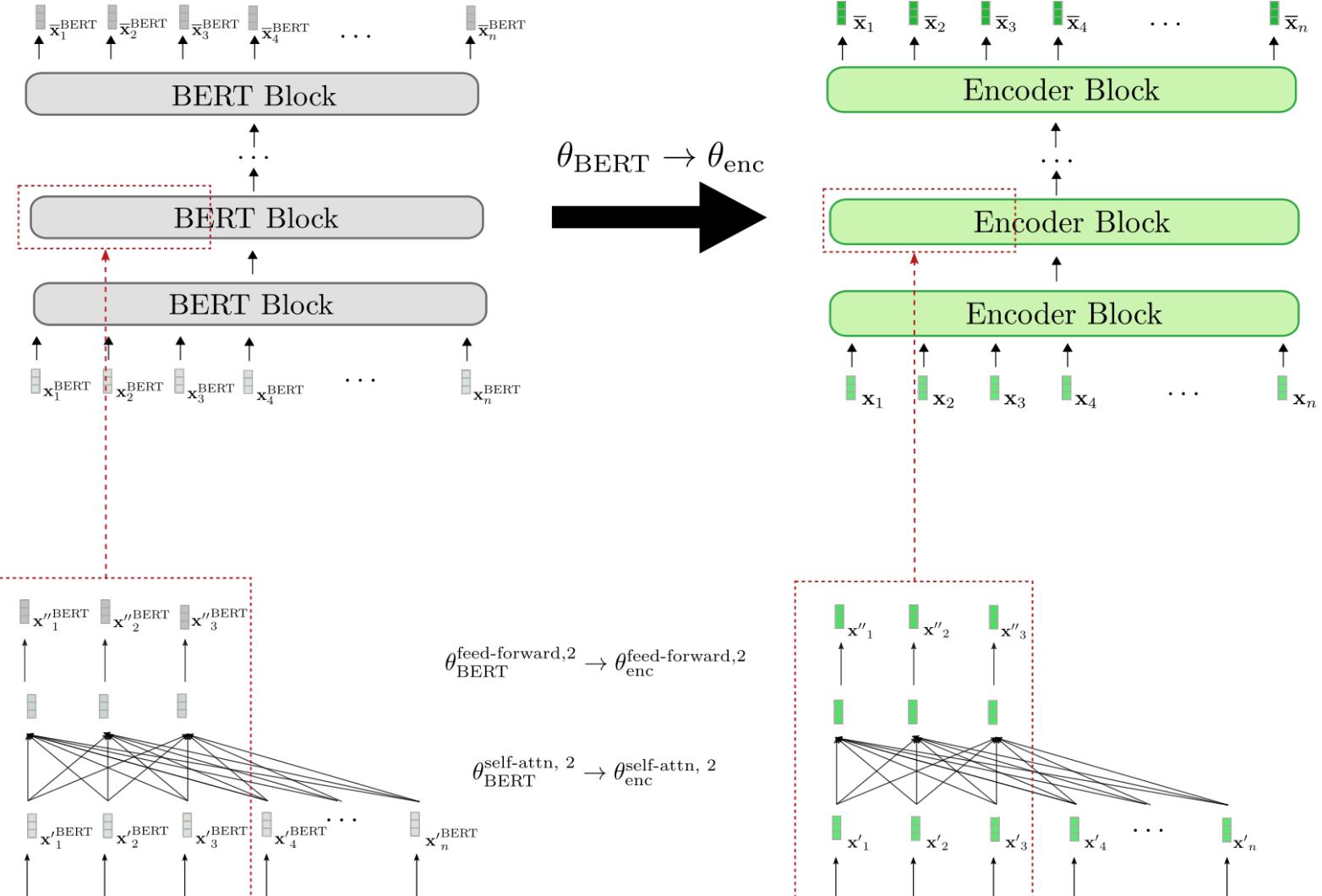


4 – Applications



Machine Translation using BERT-GPT2

BERT for Encoder

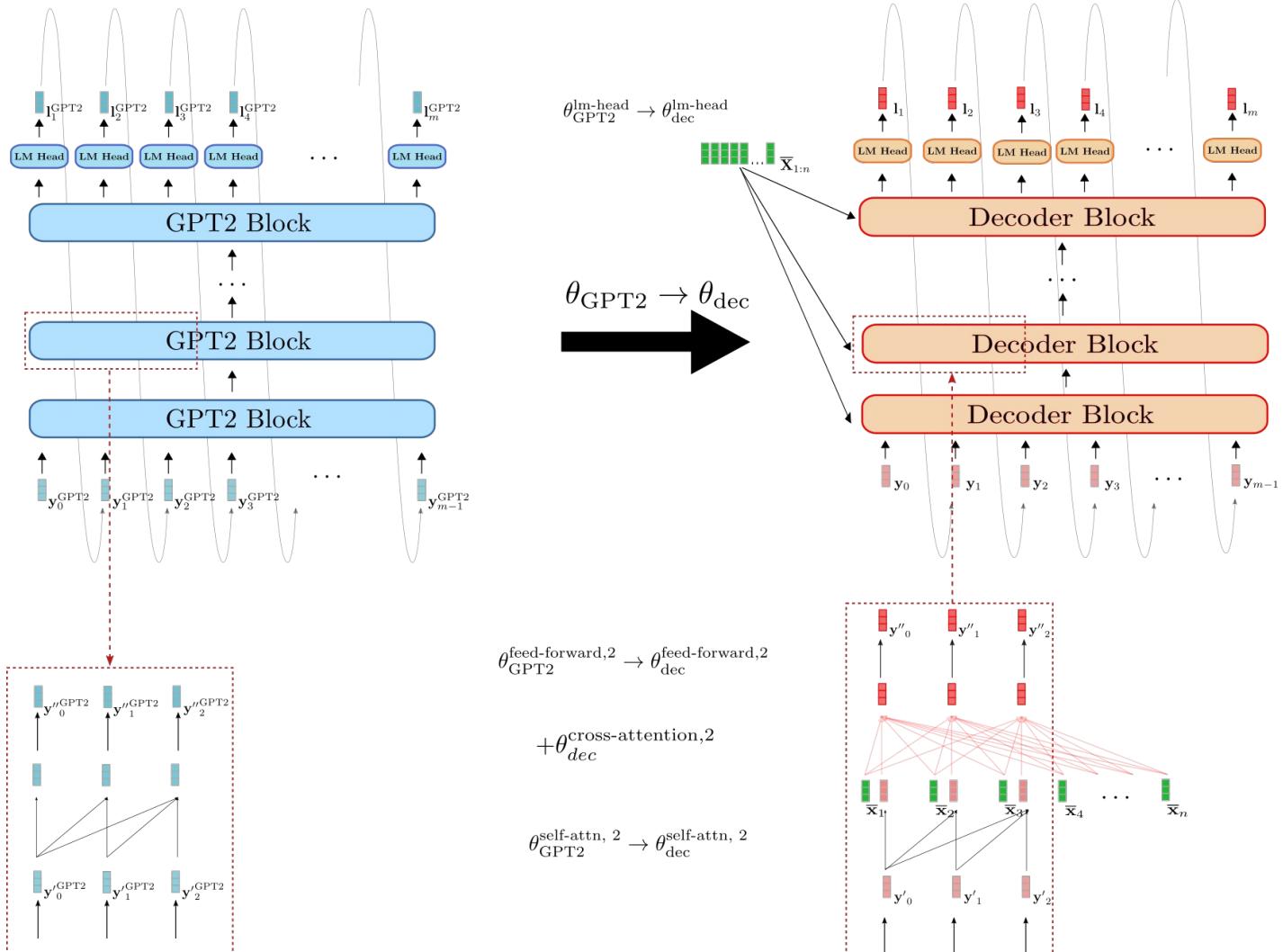


4 – Applications



Machine Translation using BERT-GPT2

GPT2 for Decoder

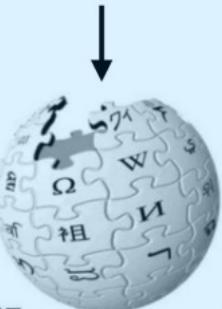


4 – Applications



Question Answering

Q: How many of Warsaw's inhabitants spoke Polish in 1933?

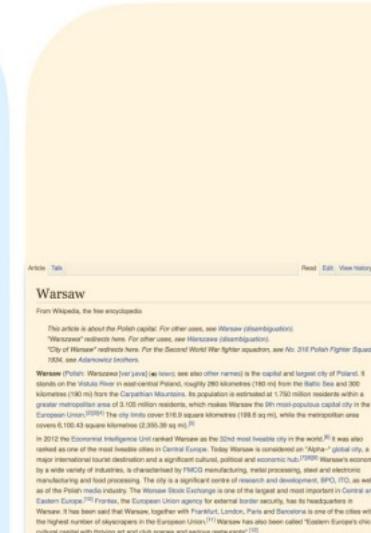


WIKIPEDIA

Document
Retriever



Information Retrieval



Document
Reader



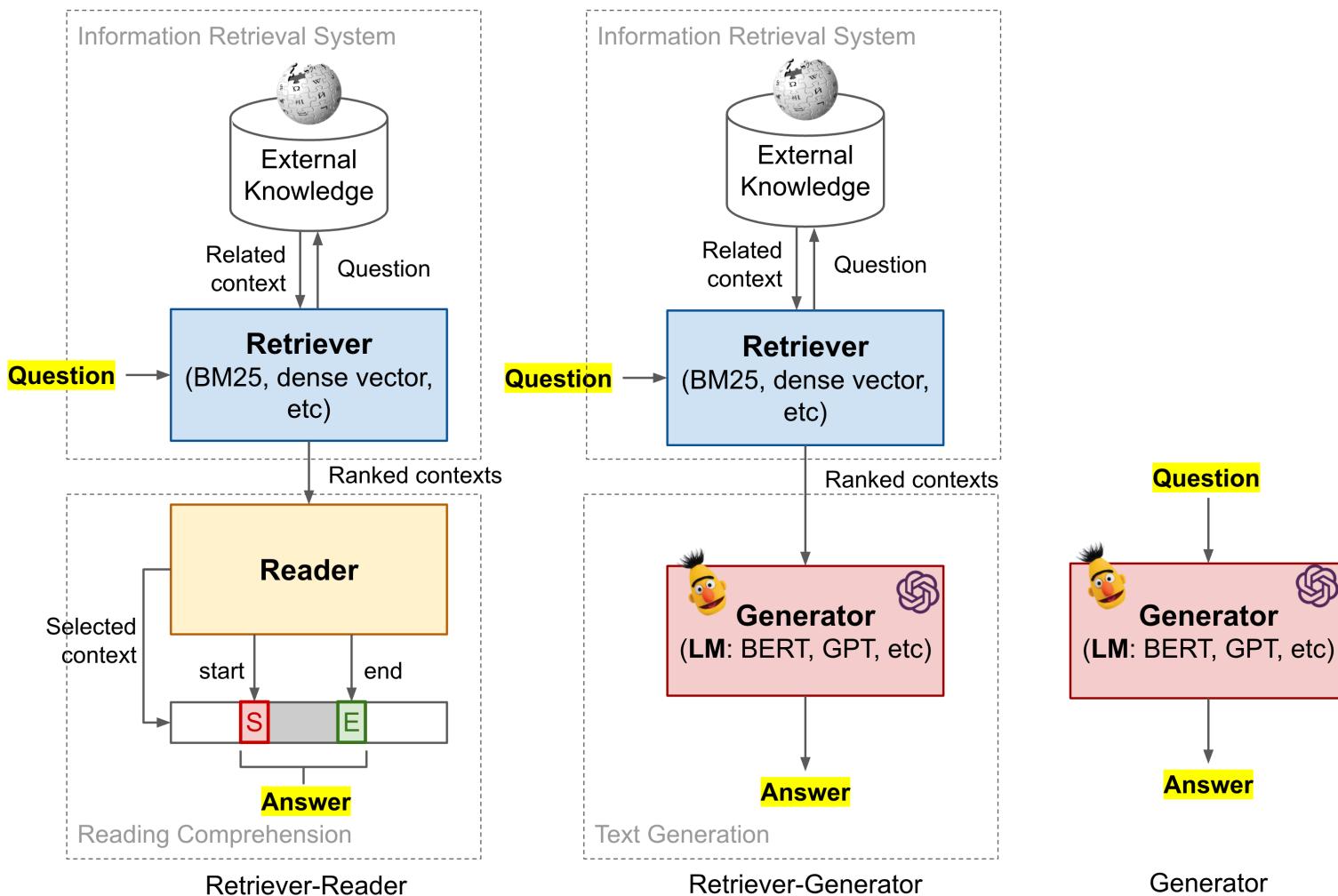
833,500

Reading Comprehension

4 – Applications



Question Answering



4 – Applications



Question Answering using BERT

- Input: A passage P - content (paragraph, document,...) and a question Q
- Output: An answer A

Article: Endangered Species Act

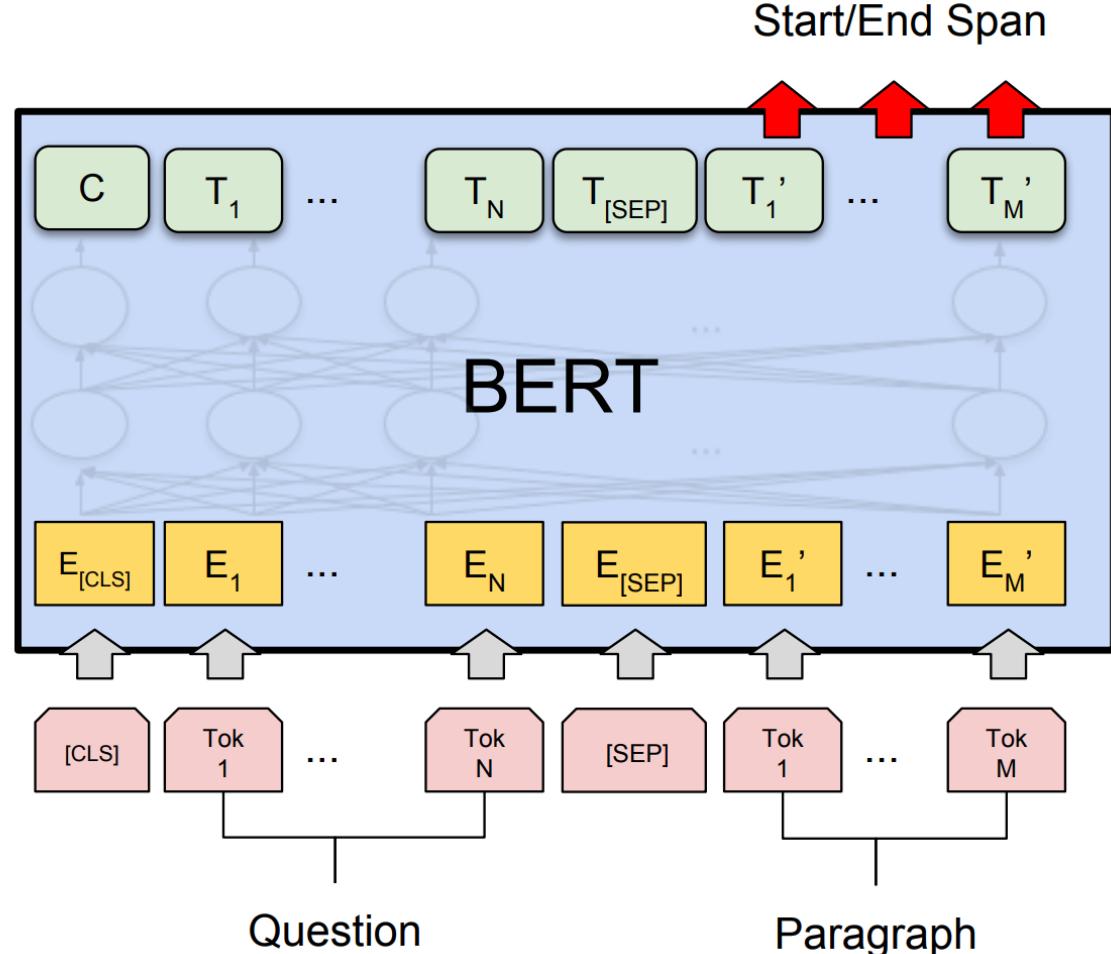
Paragraph: “*... Other legislation followed, including the Migratory Bird Conservation Act of 1929, a 1937 treaty prohibiting the hunting of right and gray whales, and the Bald Eagle Protection Act of 1940. These later laws had a low cost to society—the species were relatively rare—and little opposition was raised.*”

Question 1: “Which laws faced significant *opposition*? ”

Plausible Answer: *later laws*

Question 2: “What was the name of the 1937 *treaty*? ”

Plausible Answer: *Bald Eagle Protection Act*



4 – Applications

!

Experiment

- Source code + Checkpoint: [Link](#)

Thanks!

Any questions?