بسم الله الرحمن الرحیم

Transformer Language models

Lecture 2 - Decoder-only models

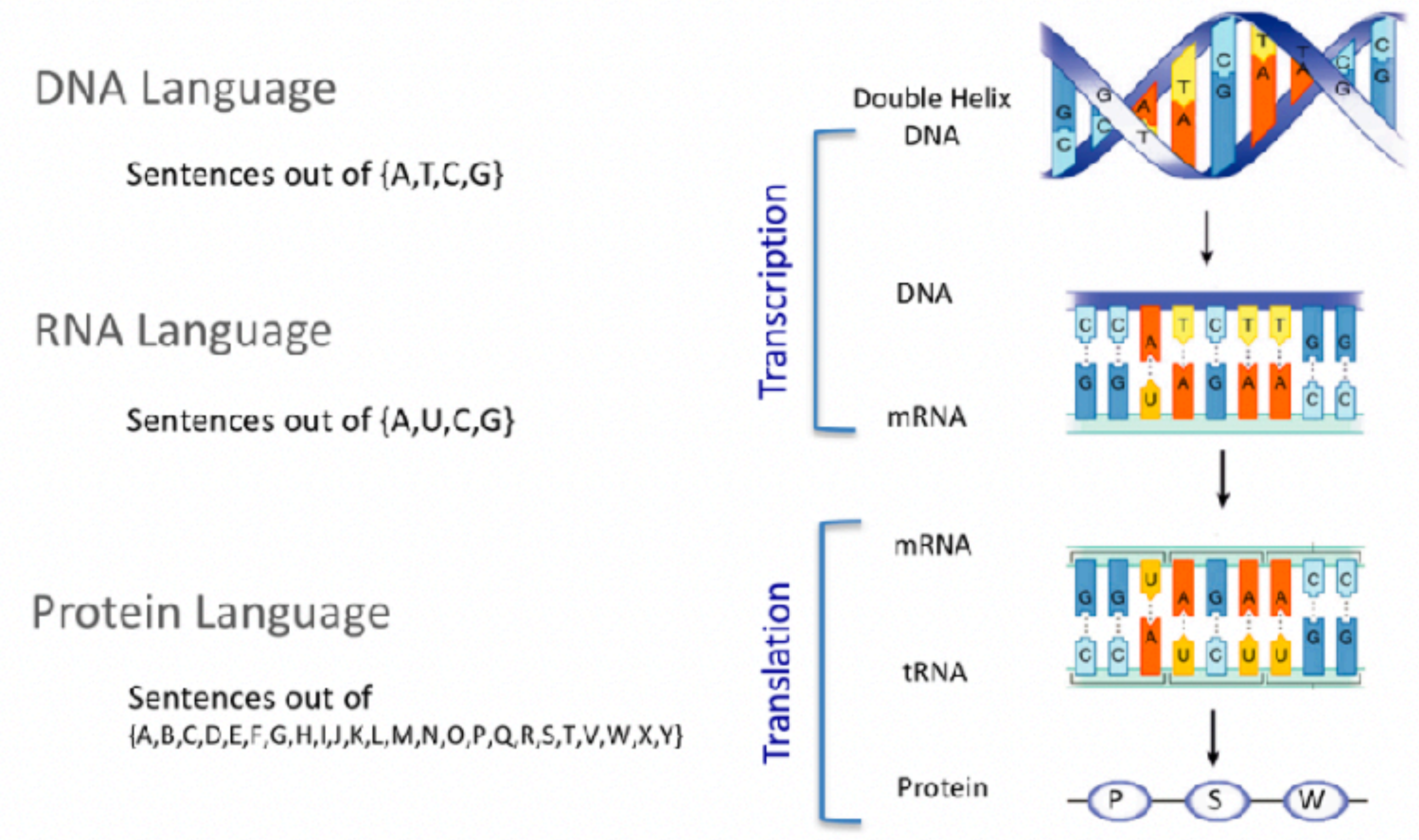
Oct. 1st 2023

Artificial Intelligence Group
Computer Engineering Department, SUT

–Language definition

Language Definition

Chomsky (1959: 137) *"A language is a collection of* ***sentences*** *of* ***finite length*** *all constructed from a* ***finite alphabet*** *(or, where our concern is limited to syntax, a finite vocabulary) of symbols."*



–Distributional Hypothesis

Distributional hypothesis


J.R. Firth

Firth (1950) *“a word is characterized by the company it keeps”*

زیادتی مطلب، کار بر خود آسان کن

صراحی می لعل و بتی چو ماهت بس


A picture of a good friendship circle in Persian culture
Made with Bing Image Creator. Powered by DALL-E

REVIEW

Language modeling

$$p(\text{start}, w_1, w_2, \ldots, w_n, \text{stop})$$

$$p(\text{start}, w_1, w_2, \ldots, w_n, \text{stop}) = \prod_{i=1}^{n+1} \gamma(w_i \mid w_1, w_2, \ldots, w_{i-1})$$

REVIEW

N–gram Language modeling

$$p(\text{start}, w_1, w_2, \ldots, w_n, \text{stop}) = \prod_{i=1}^{n+1} \gamma(w_i \mid w_1, w_2, \ldots, w_{i-1})$$

Bi-gram

$$p(\text{start}, w_1, w_2, \ldots, w_n, \text{stop}) = \prod_{i=1}^{n+1} \gamma(w_i \mid w_{i-1})$$

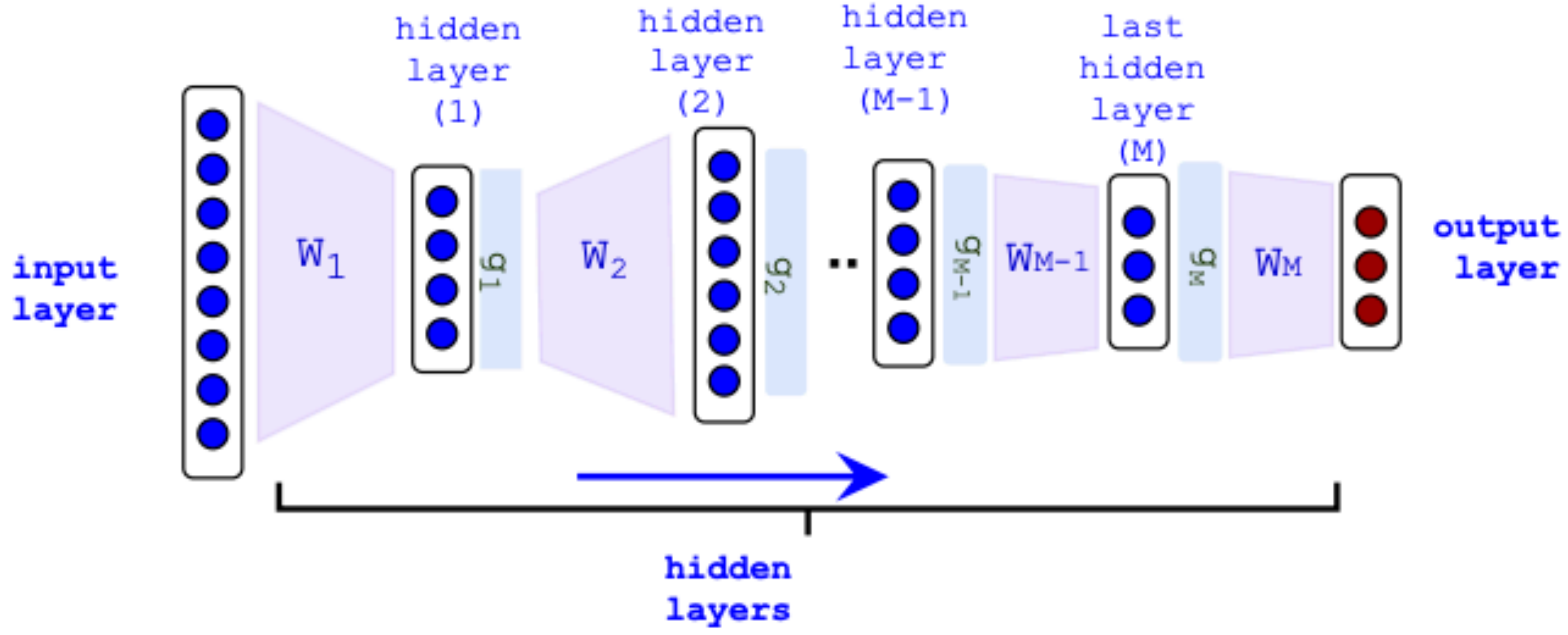
Markov (m$^{\text{th}}$ order)

$$p(\text{start}, w_1, w_2, \ldots, w_n, \text{stop}) = \prod_{i=1}^{n+1} \gamma(w_i \mid w_{i-m}, \ldots, w_{i-1})$$

REVIEW

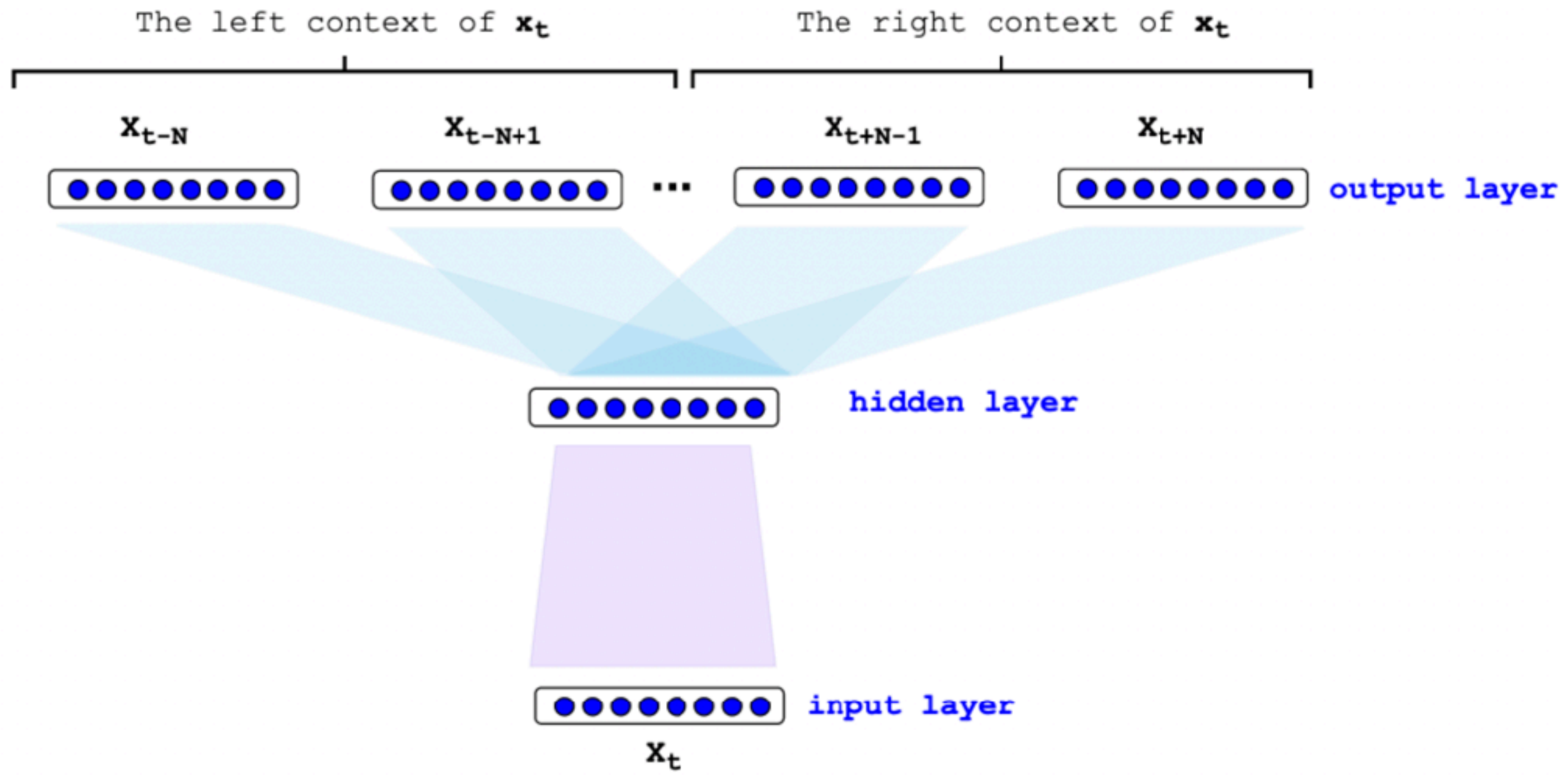
Neural Language modeling

$$p(\text{start}, w_1, w_2, \ldots, w_n, \text{stop})$$



REVIEW

Skip–gram – Similar to Language Models



$$\sum_{t=1}^{M} \sum_{c \in [t-N, t+N]} \log p(w_c \mid w_t)$$

$$p(w_c \mid w_t; \theta) = \frac{e^{v_c \cdot v_t}}{\sum_{c' \in C} e^{v_{c'} \cdot v_t}}$$

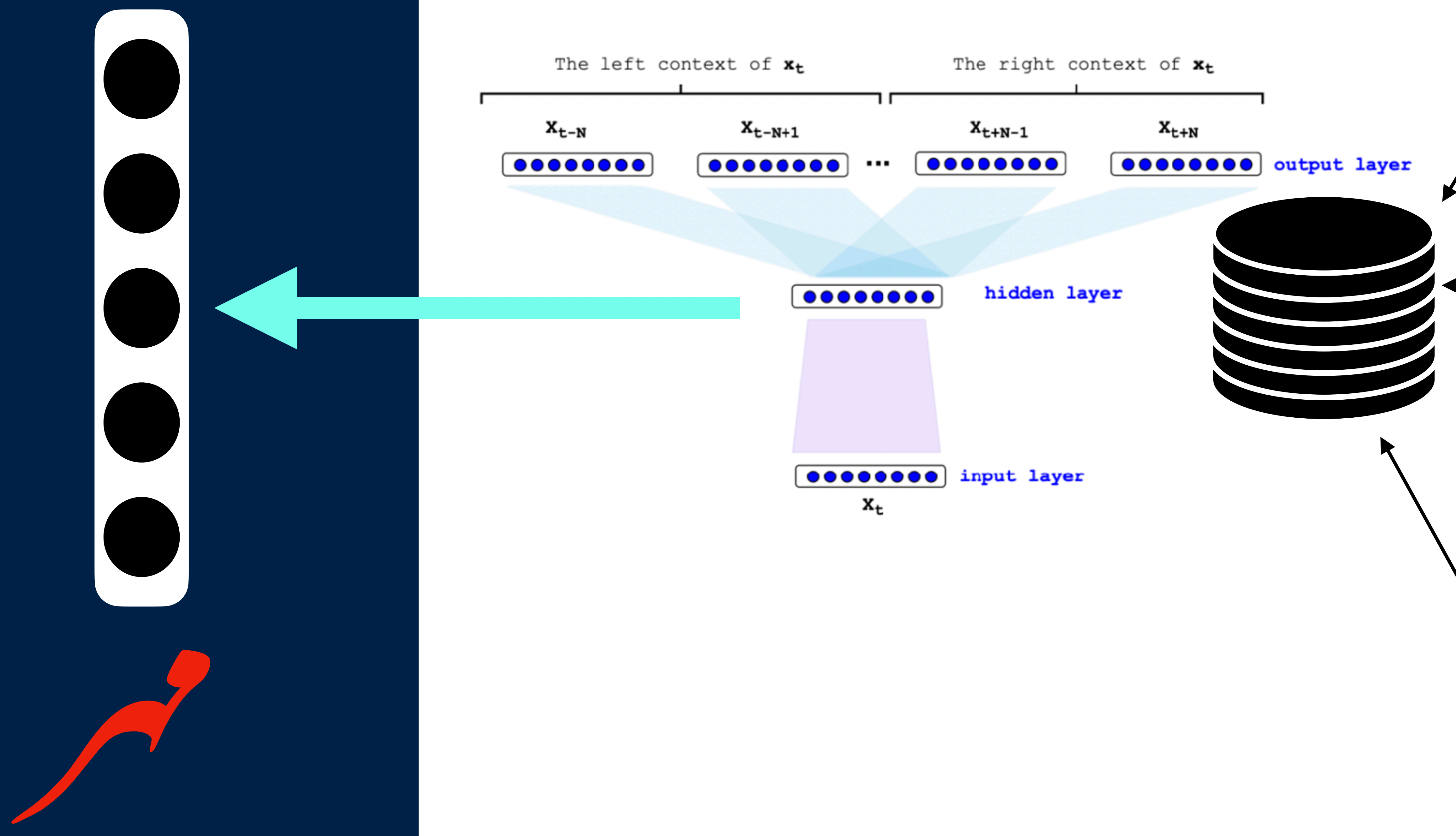
Maximizing the following likelihood:

$$\sum_{t=1}^{T} \left[\sum_{c \in [t-N, t+N]} \log\left(1 + e^{-s(w_t,\ w_c)}\right) + \sum_{w_r \in \mathcal{N}_{t,c}} \log\left(1 + e^{s(w_t,\ w_r)}\right) \right]$$

$$s(w_t,\ w_c) = {v_t}^{\top} \cdot v_c$$

Fixed embeddings – Skip-gram

REVIEW



...که دگر نه عشق خورشید و نه مهر ماه دارم

فروزنده ماه و ناهید و مهر...

...فرو شست از نگار و نقش ماه مهر و آبانش

Fixed embeddings – Skip-gram

آبان
ماه
مهر
خورشید
ناهید

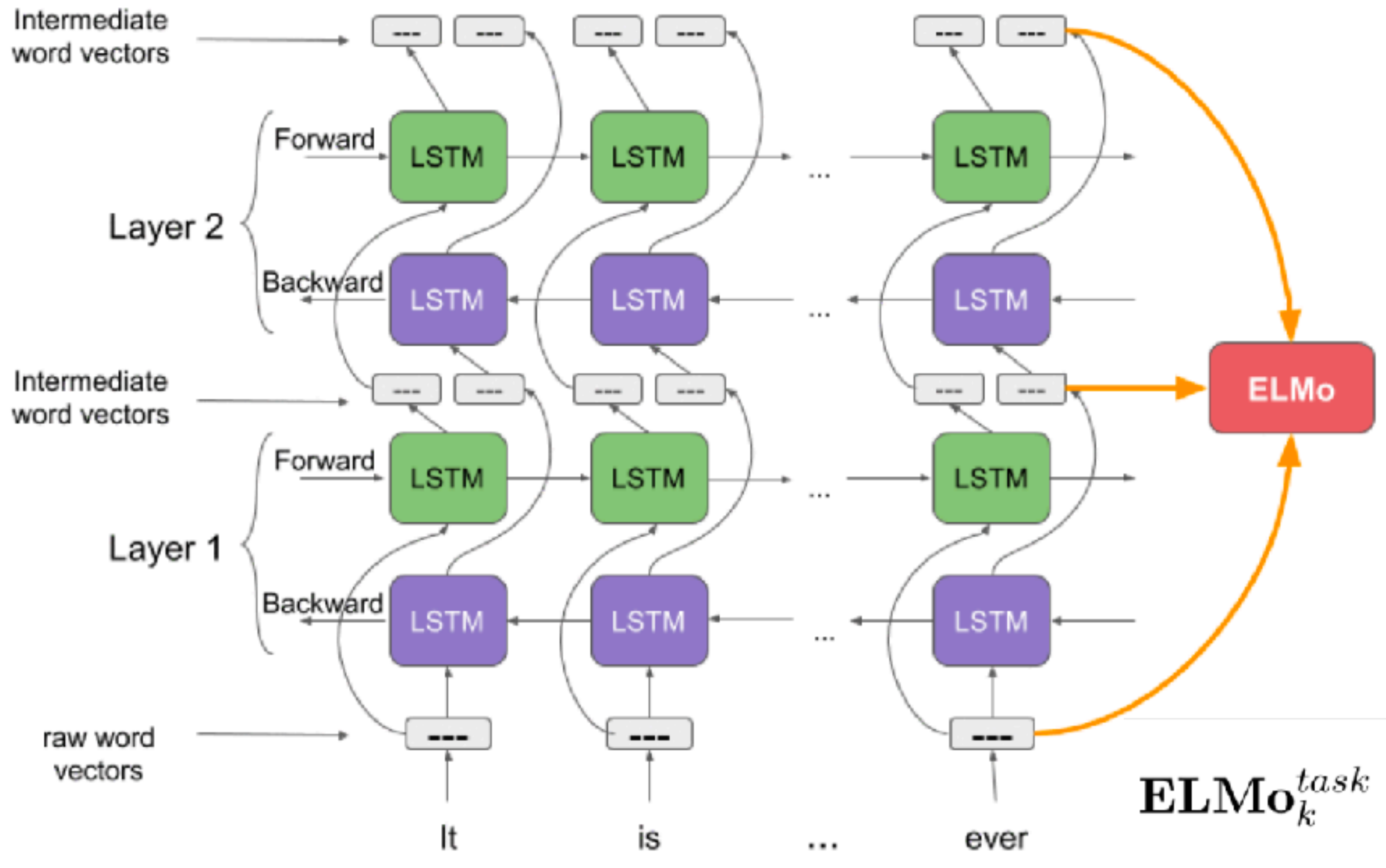
... که دگر نه عشق خورشید و نه مهر ماه دارم

فروزنده ماه و ناهید و مهر ...

... فرو شست از نگار و نقش ماه مهر و آبانش

REVIEW

ELMO: Deep contextualized word representations



$$\mathbf{ELMo}_k^{task} = E(R_k;\Theta^{task}) = \gamma^{task}\sum_{j=0}^{L} s_j^{task}\mathbf{h}_{k,j}^{LM}$$

ELMo (Peters et al., 2018; NAACL 2018 best paper)

- Train two separate unidirectional LMs (left-to-right and right-to-left) based on LSTMs
- Feature-based approach: pre-trained representations used as input to task-specific models

– Self-attention

آبان
ماه
مهر
خورشید
ناهید

How to contextualize the fixed embeddings?

...که دگر نه عشق خورشید و نه مهر ماه دارم

فروزنده ماه و ناهید و مهر...

...فرو شست از نگار و نقش ماه مهر و آبانش

Attention

Self-Attention Idea

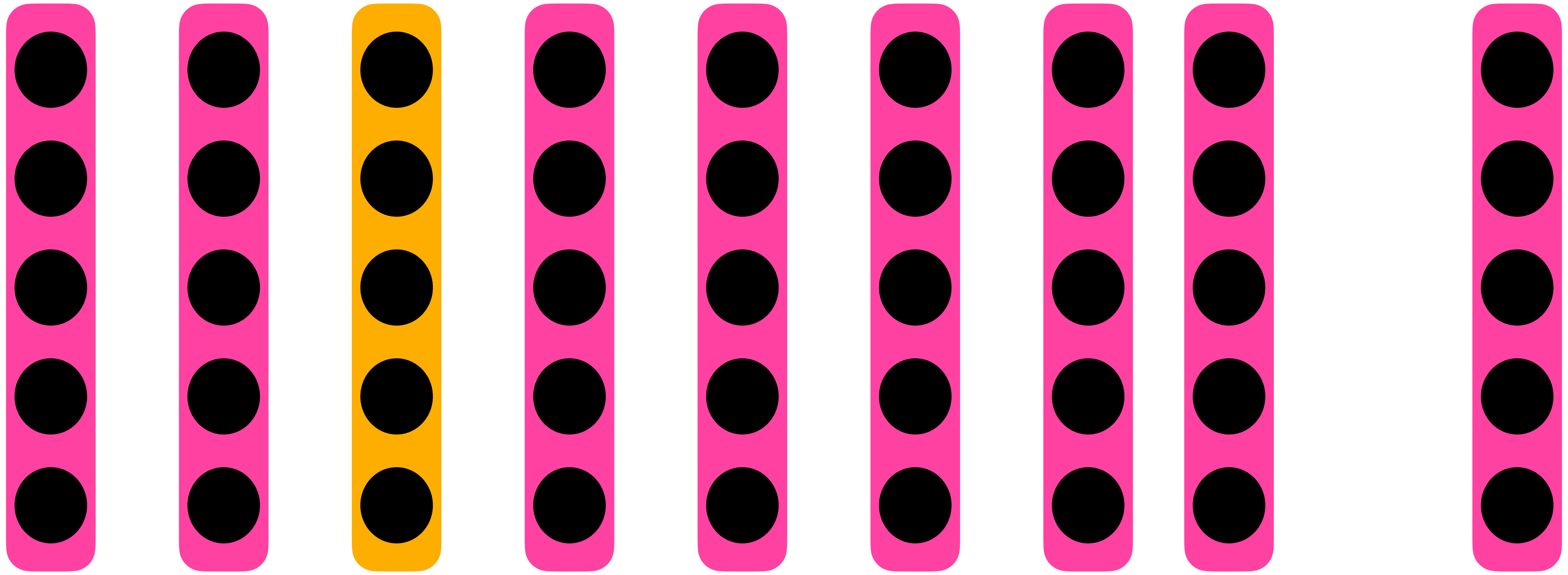
Input embeddings

$$x_1, x_2, \ldots, x_n$$

Output embeddings

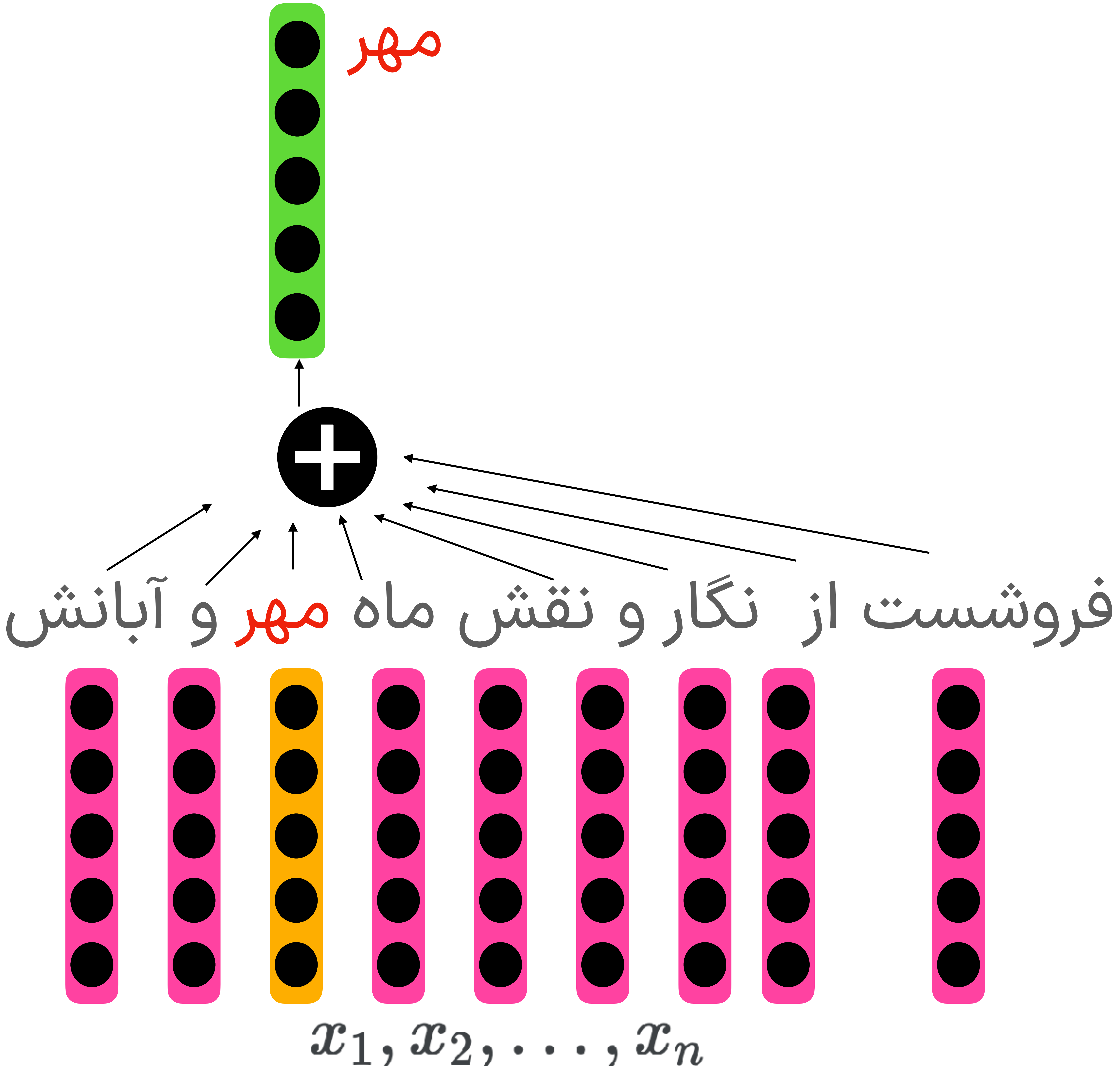
$$y_1, y_2, \ldots, y_n$$

فرو شست از نگار و نقش ماه مهر و آبانش



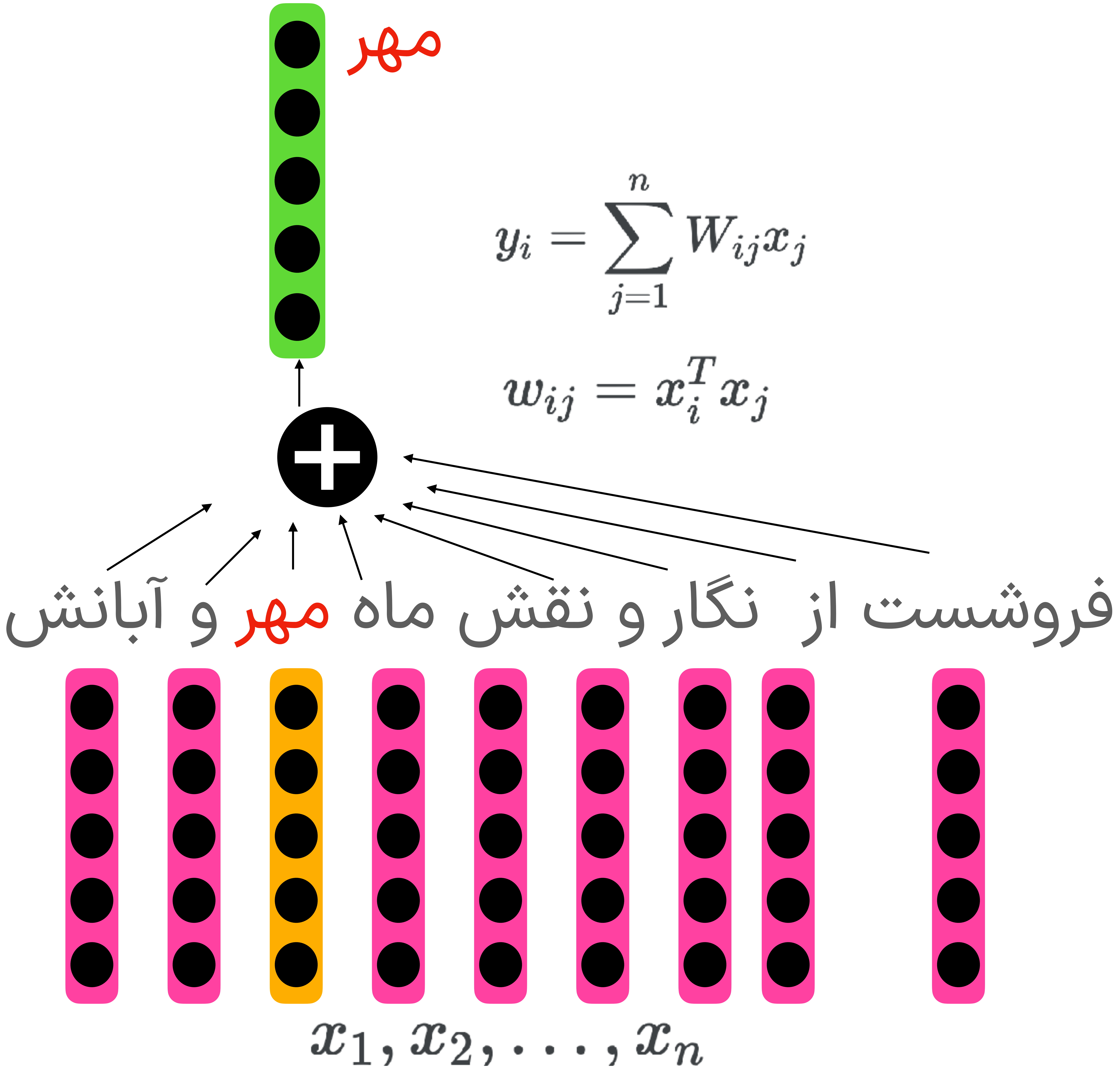
Attention

Self-Attention Idea



Self-Attention

Attention

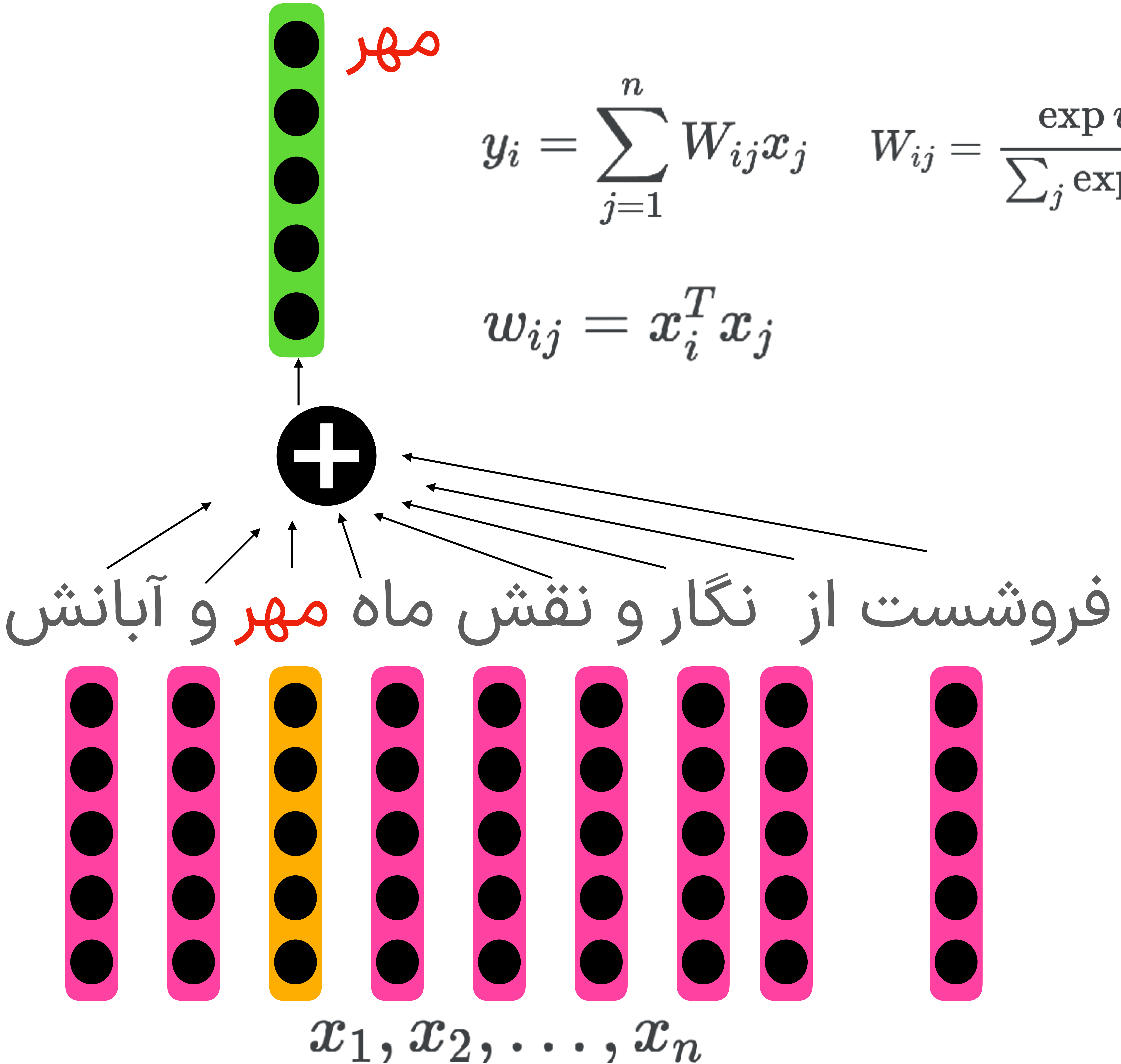


$$y_i = \sum_{j=1}^{n} W_{ij} x_j$$

$$w_{ij} = x_i^T x_j$$

Self–Attention

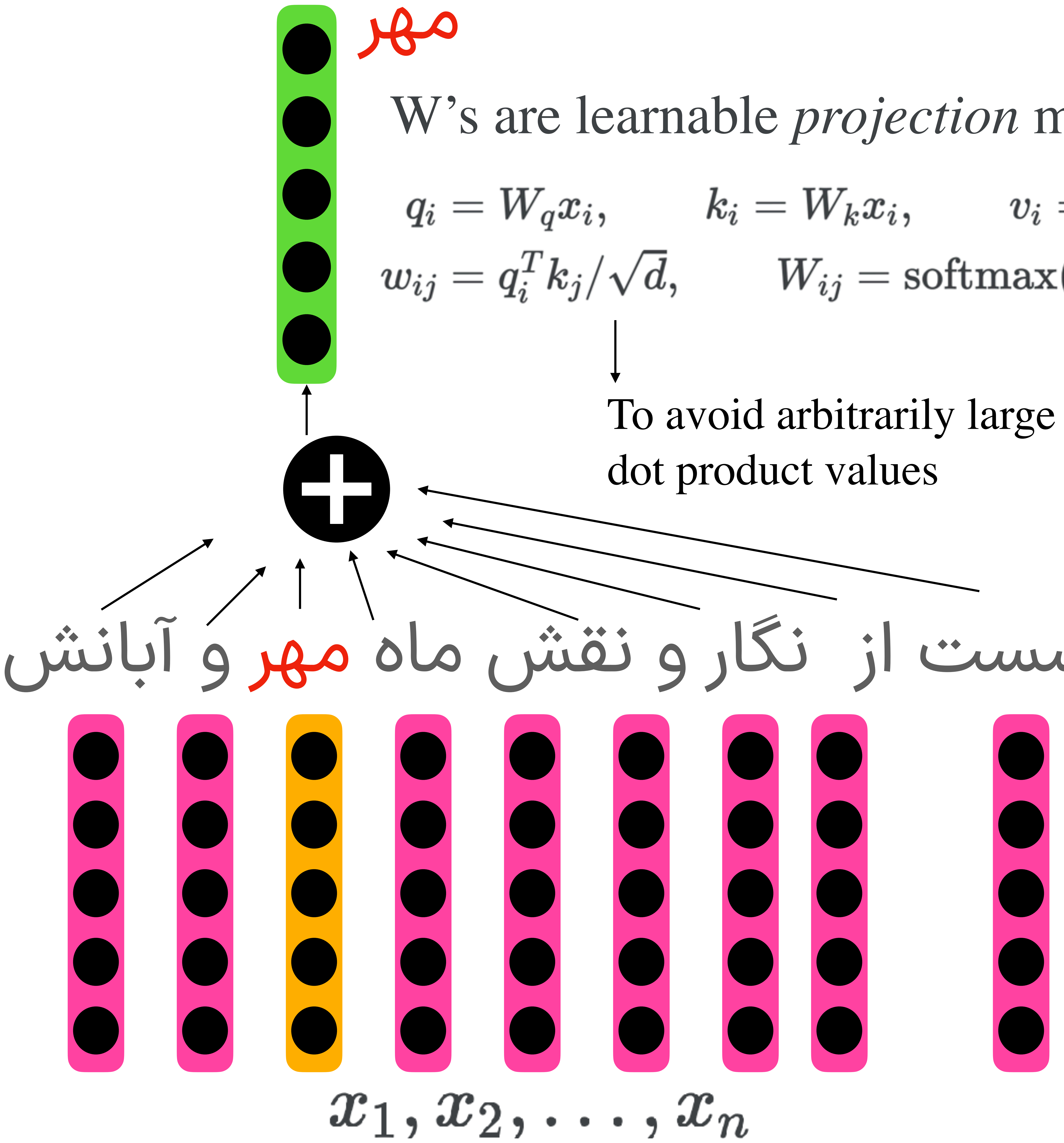
Attention



$$y_i = \sum_{j=1}^{n} W_{ij} x_j \qquad W_{ij} = \frac{\exp w_{ij}}{\sum_j \exp w_{ij}}$$

$$w_{ij} = x_i^T x_j$$

Attention



W's are learnable *projection* matrices

$$q_i = W_q x_i, \qquad k_i = W_k x_i, \qquad v_i = W_v x_i$$

$$w_{ij} = q_i^T k_j / \sqrt{d}, \qquad W_{ij} = \text{softmax}(w_{ij}), \qquad y_i = \sum_j W_{ij} v_j.$$

To avoid arbitrarily large (positive or negative) dot product values

Attention

Attention

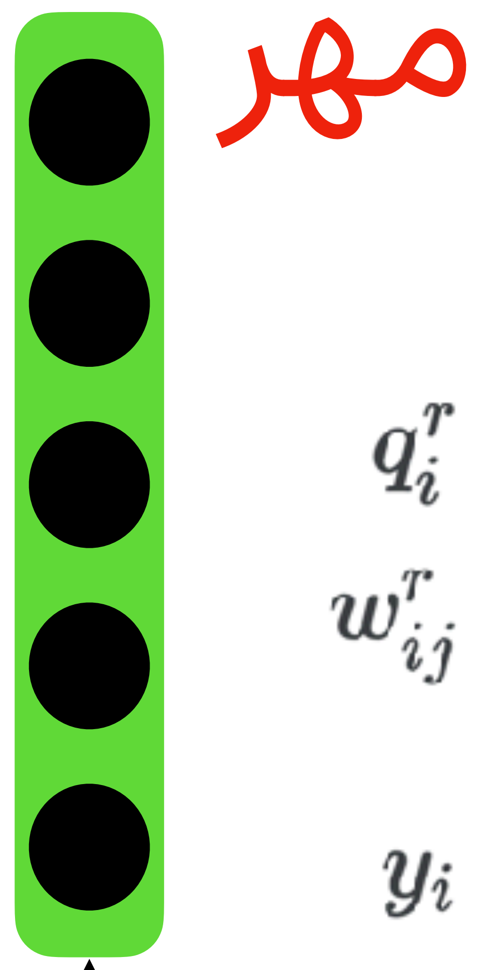
- Inputs: a query q and a set of key-value (k-v) pairs to an output
  - All presented as vectors

- Output is weighted sum of values

- Weight of each value: inner product of query and corresponding key

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

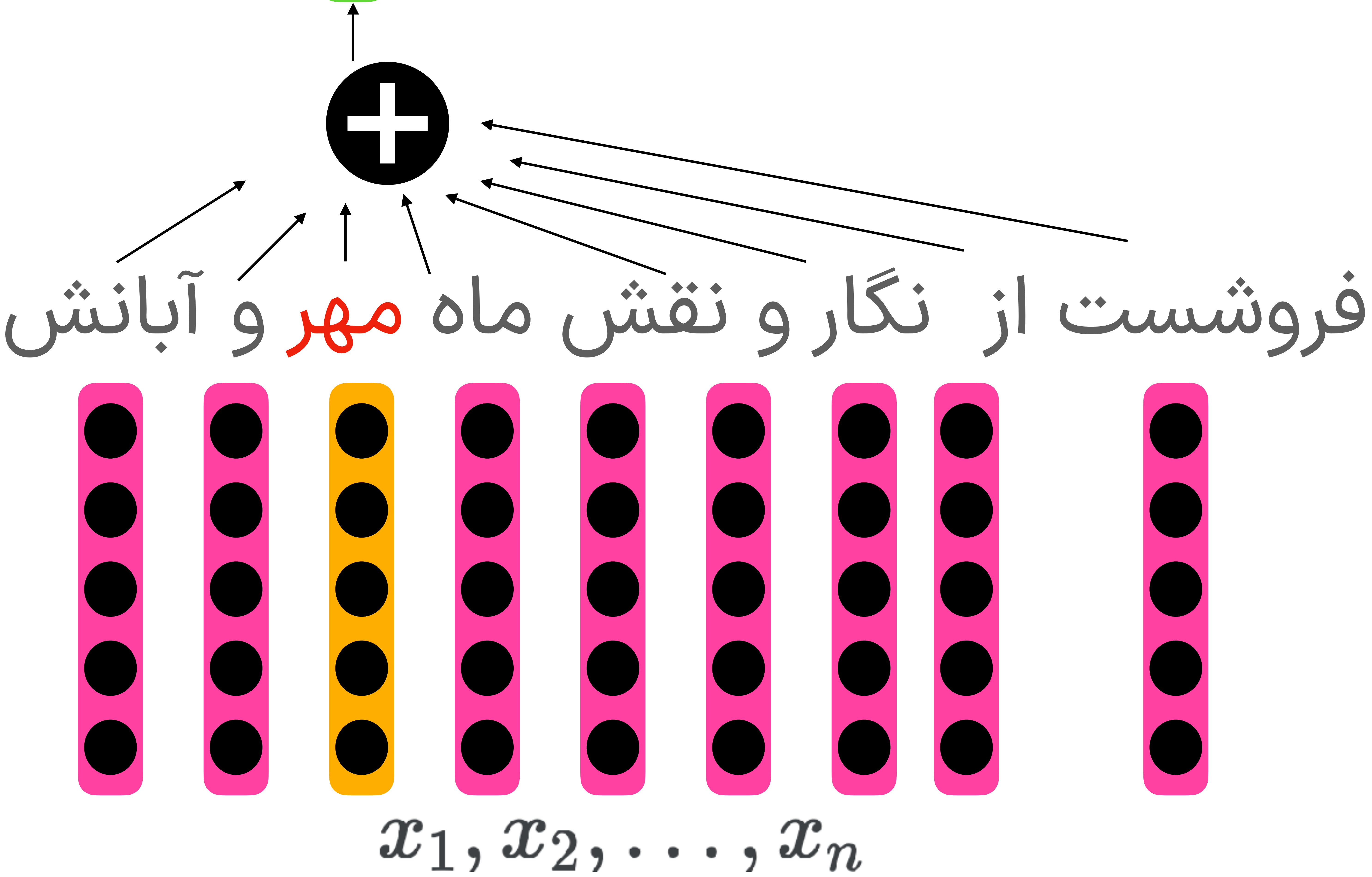
Multihead Attention

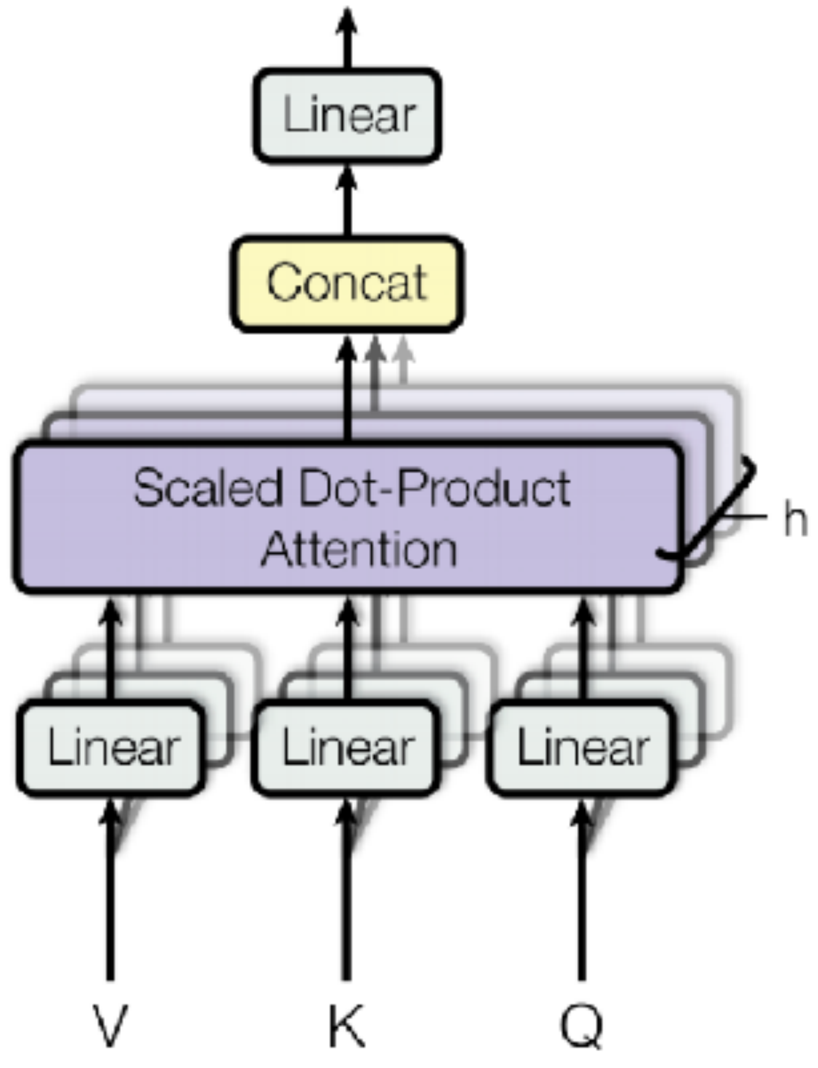


$$q_i^r = W_q^r x_i, \qquad k_i^r = W_k^r x_i, \qquad v_i^r = W_v^r x_i$$

$$w_{ij}^r = \langle q_i^r, k_j^r \rangle / \sqrt{d}, \qquad W_{ij}^r = \mathrm{softmax}(w_{ij}^r), \qquad y_i^r = \sum_j W_{ij}^r v_j,$$

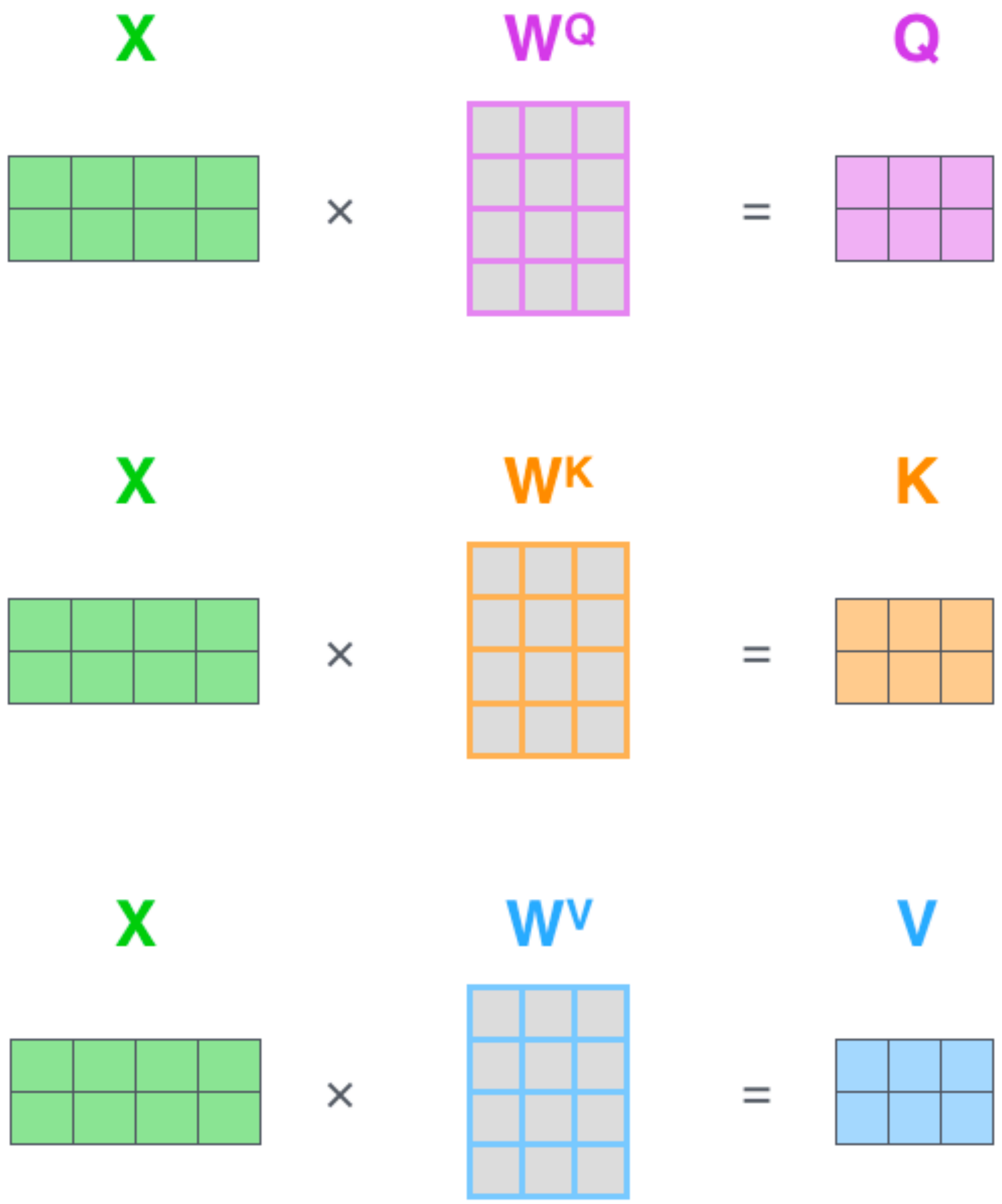
$$y_i = W_y \mathrm{concat}[y_i^1, y_i^2, \ldots].$$





Attention

Matrix Attention



Attention

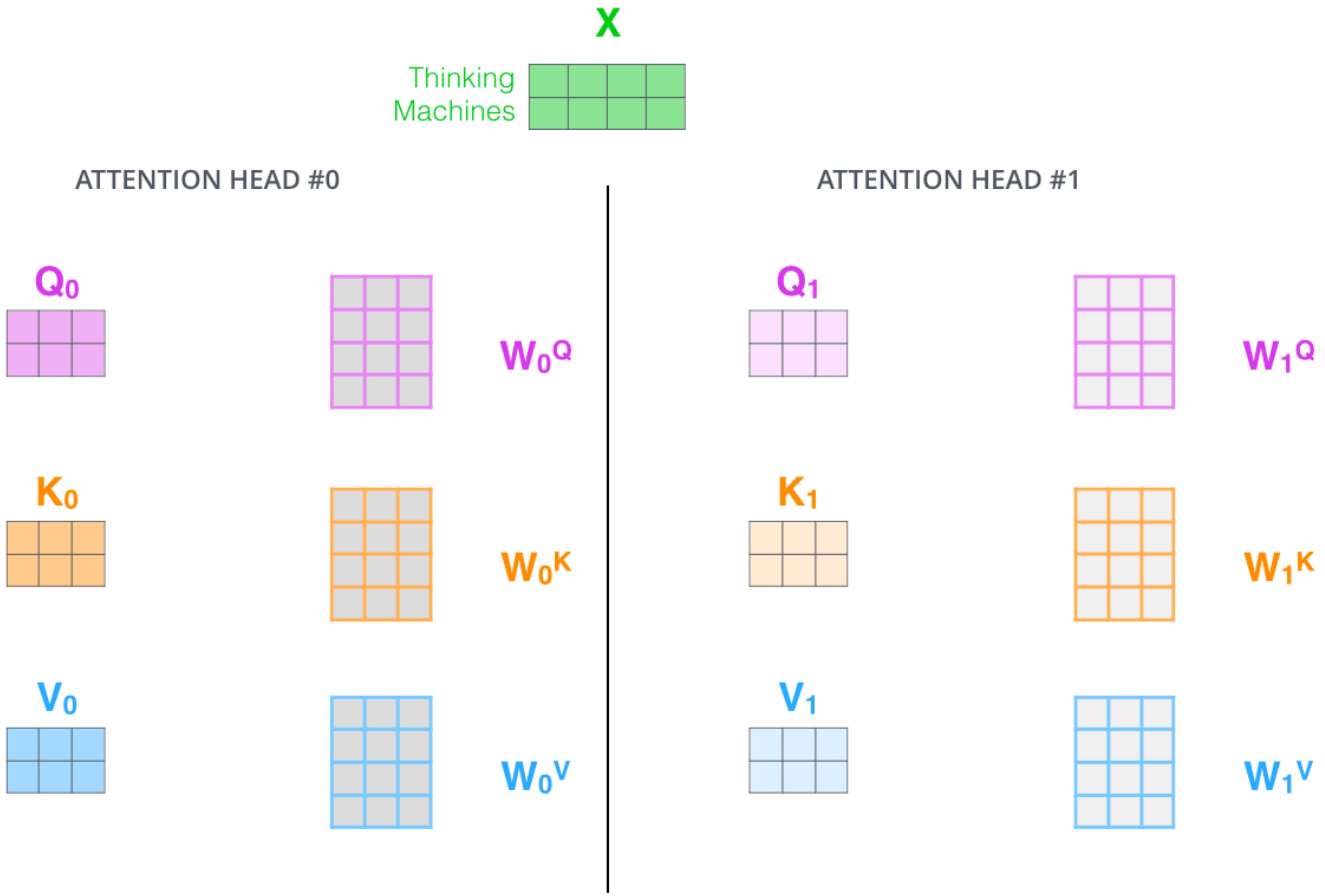
Matrix Attention

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

$$= Z$$

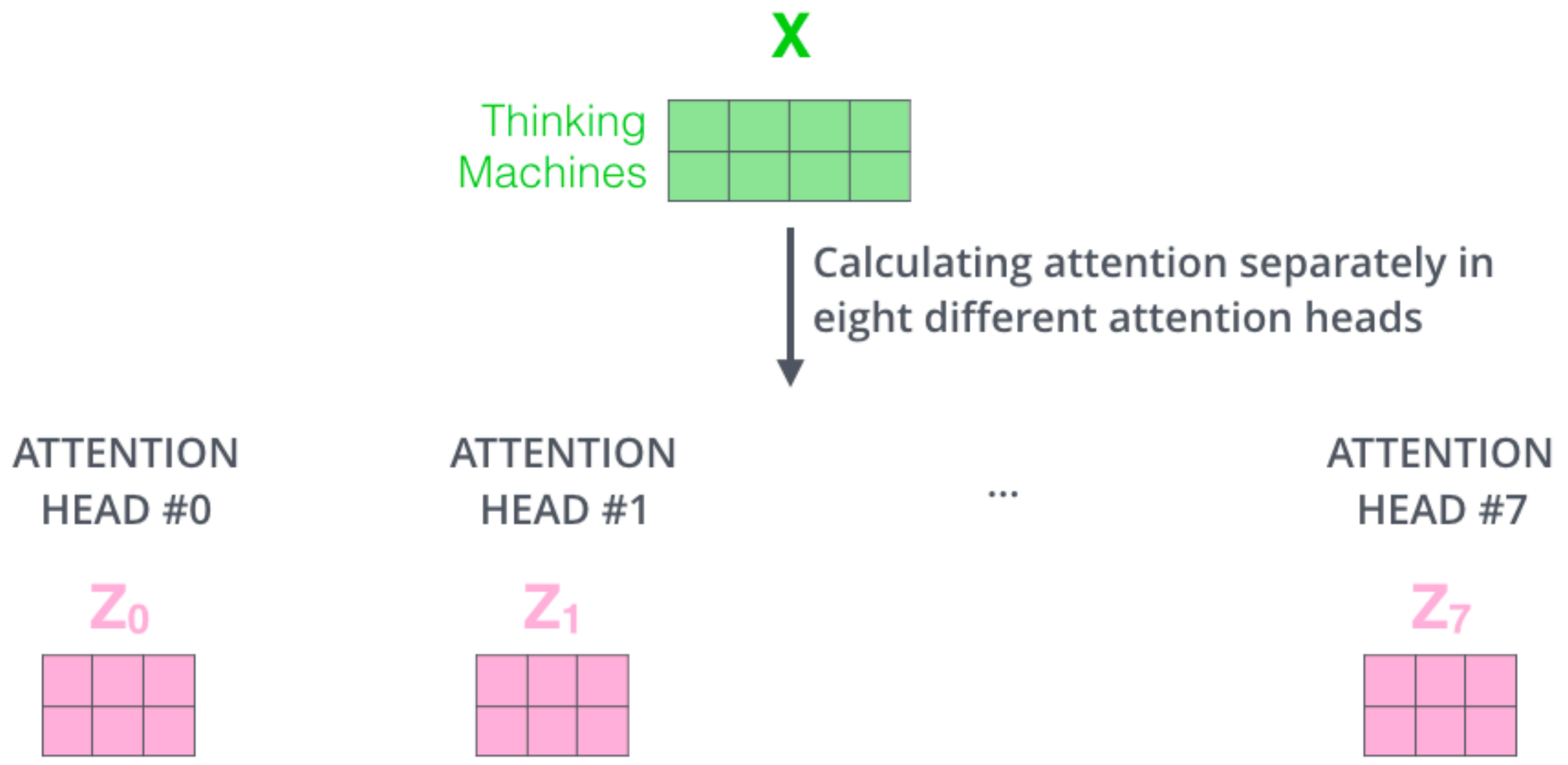
Attention

Matrix Attention



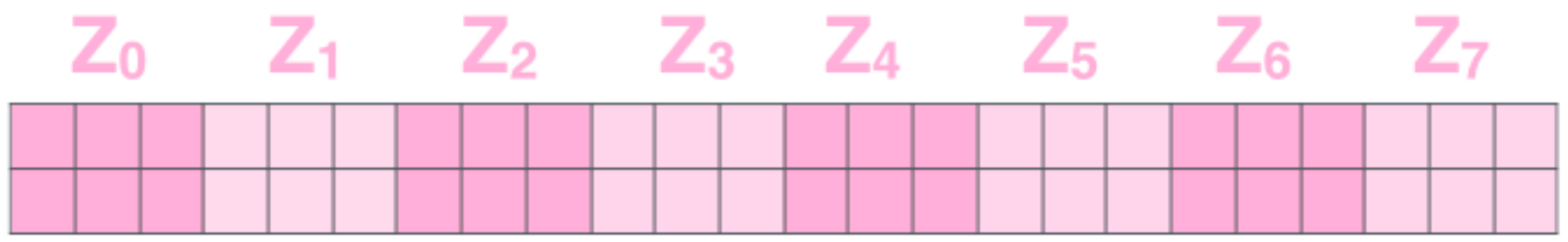
Attention

Matrix Attention



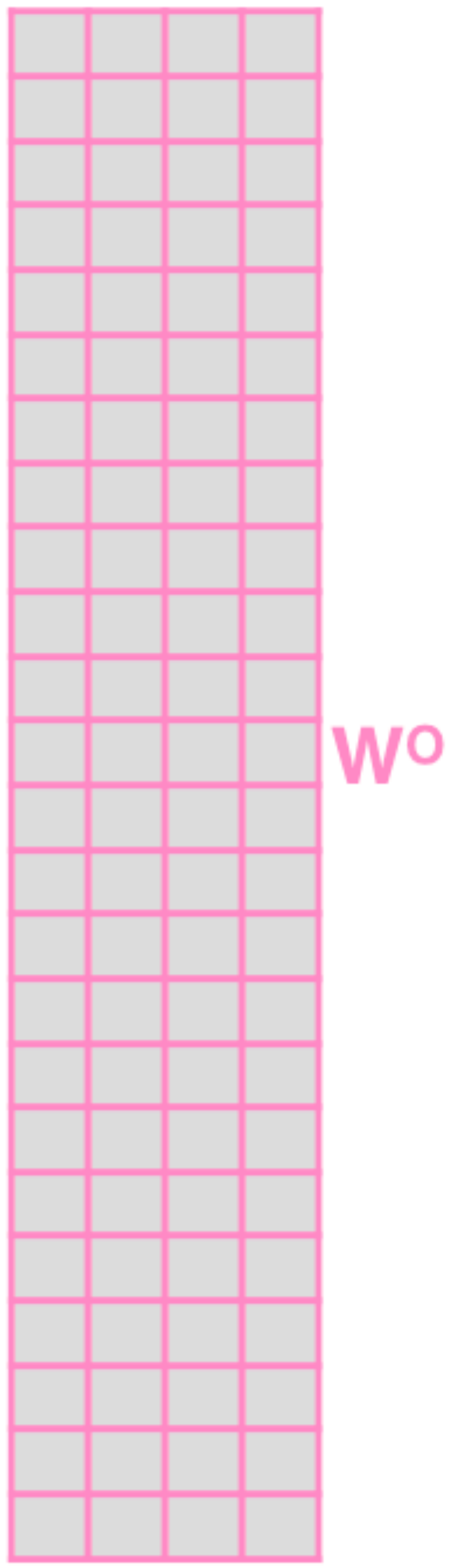
Attention

1) Concatenate all the attention heads



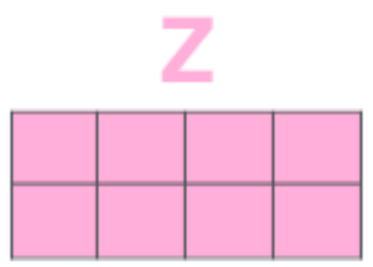
2) Multiply with a weight matrix W^O that was trained jointly with the model

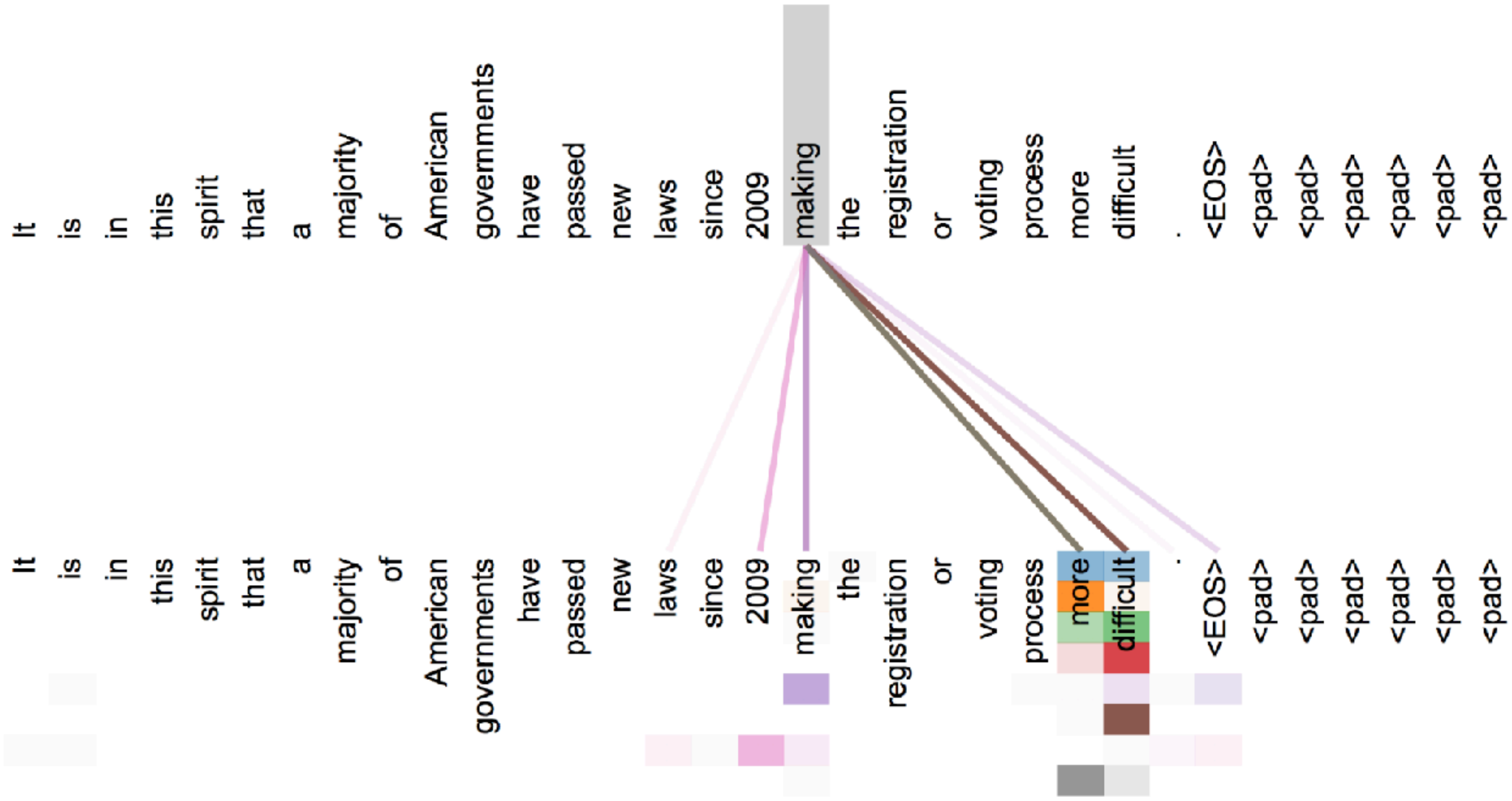
X



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

=




It is in this spirit that a majority of American governments have passed new laws since 2009 making the registration or voting process more difficult . <EOS> <pad> <pad> <pad> <pad> <pad> <pad>
It is in this spirit that a majority of American governments have passed new laws since 2009 making the registration or voting process more difficult . <EOS> <pad> <pad> <pad> <pad> <pad> <pad>

– Transformers

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

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

https://arxiv.org/abs/1706.03762

–Next lecture

بسم الله الرحمن الرحیم

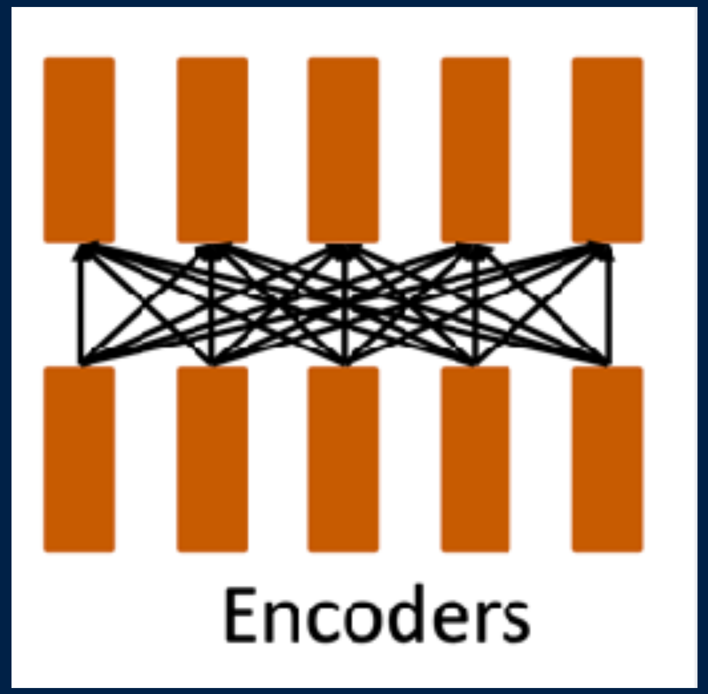
Transformer Language models

Lecture 3 - Transformers (ii)

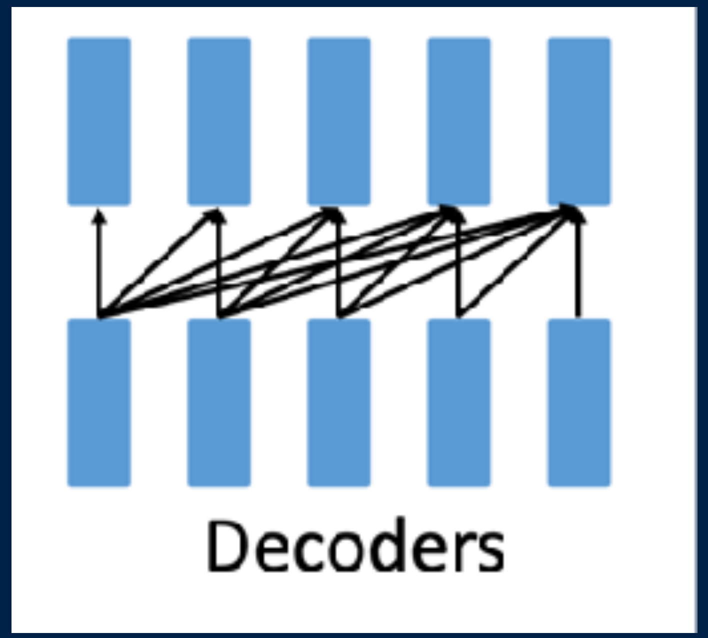
Oct. 10th 2023

Artificial Intelligence Group
Computer Engineering Department, SUT

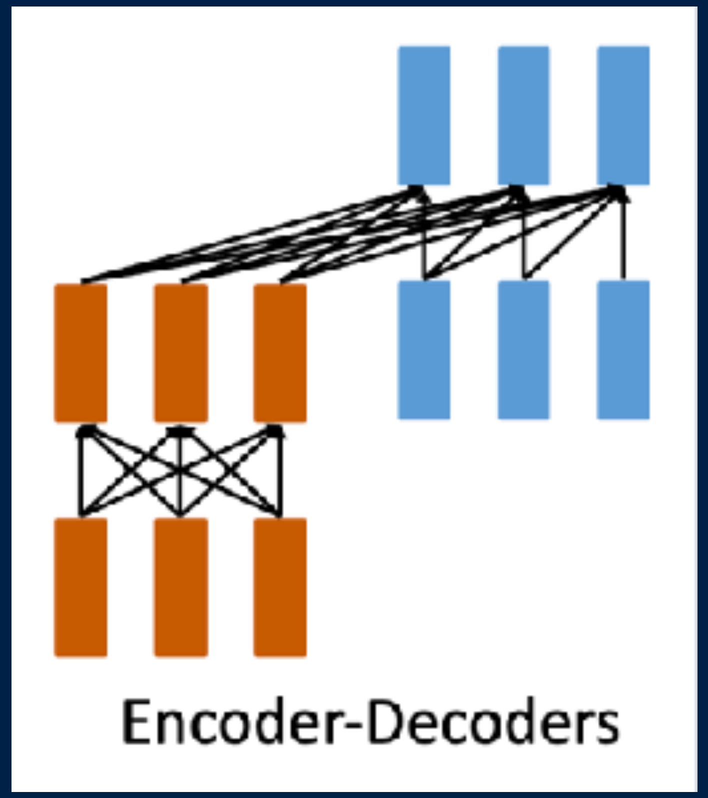
Transformer Architectures



- Encoder-only (e.g., BERT): bidirectional contextual embeddings



- Decoder-only (e.g., GPT-x): unidirectional contextual embeddings, generate one token at a time



- Encoder-decoder (e.g., T5): encode input, decode output

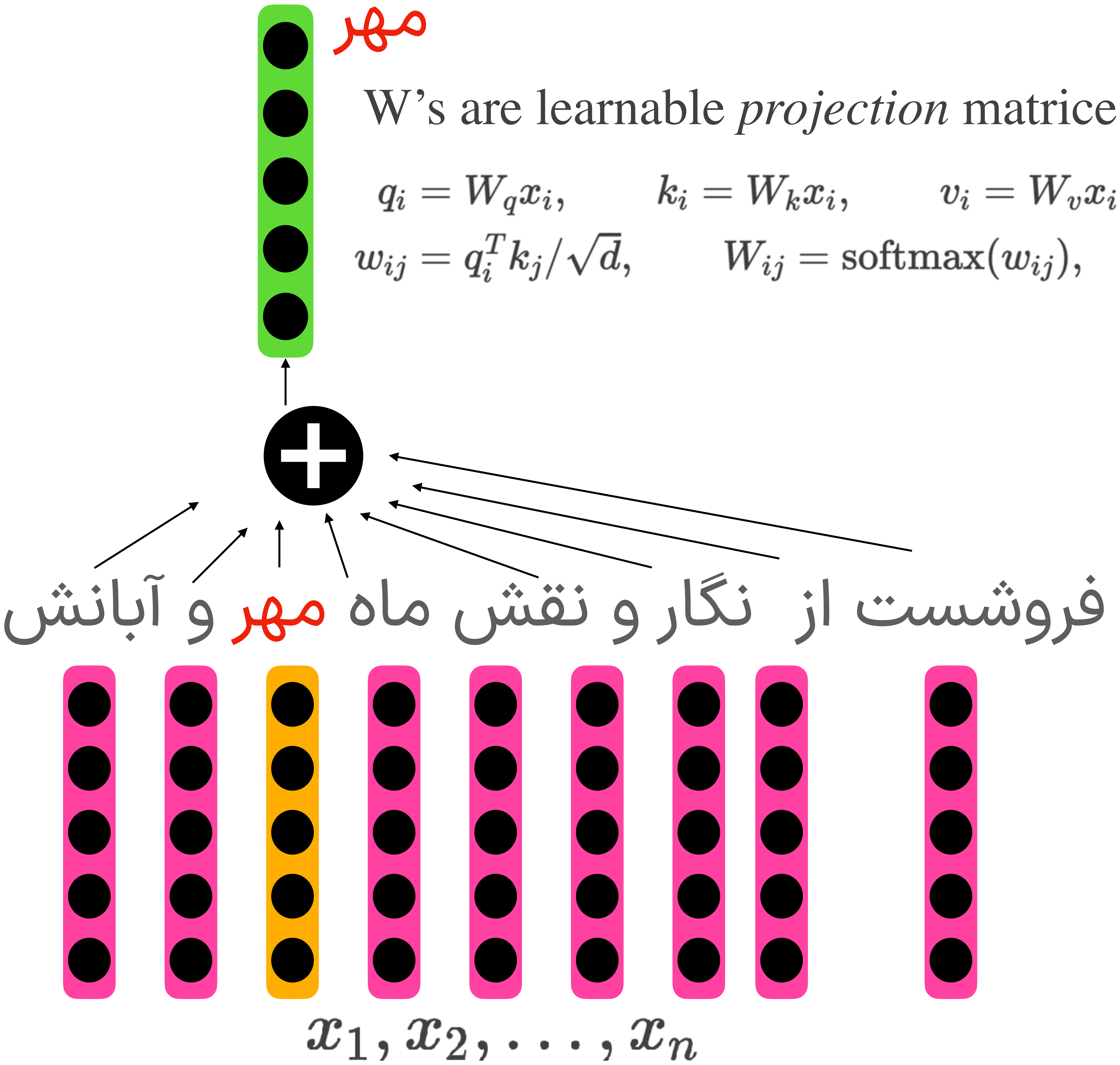
Attention

REVIEW

W's are learnable *projection* matrices

$$q_i = W_q x_i, \qquad k_i = W_k x_i, \qquad v_i = W_v x_i$$

$$w_{ij} = q_i^T k_j / \sqrt{d}, \qquad W_{ij} = \mathrm{softmax}(w_{ij}), \qquad y_i = \sum_j W_{ij} v_j.$$



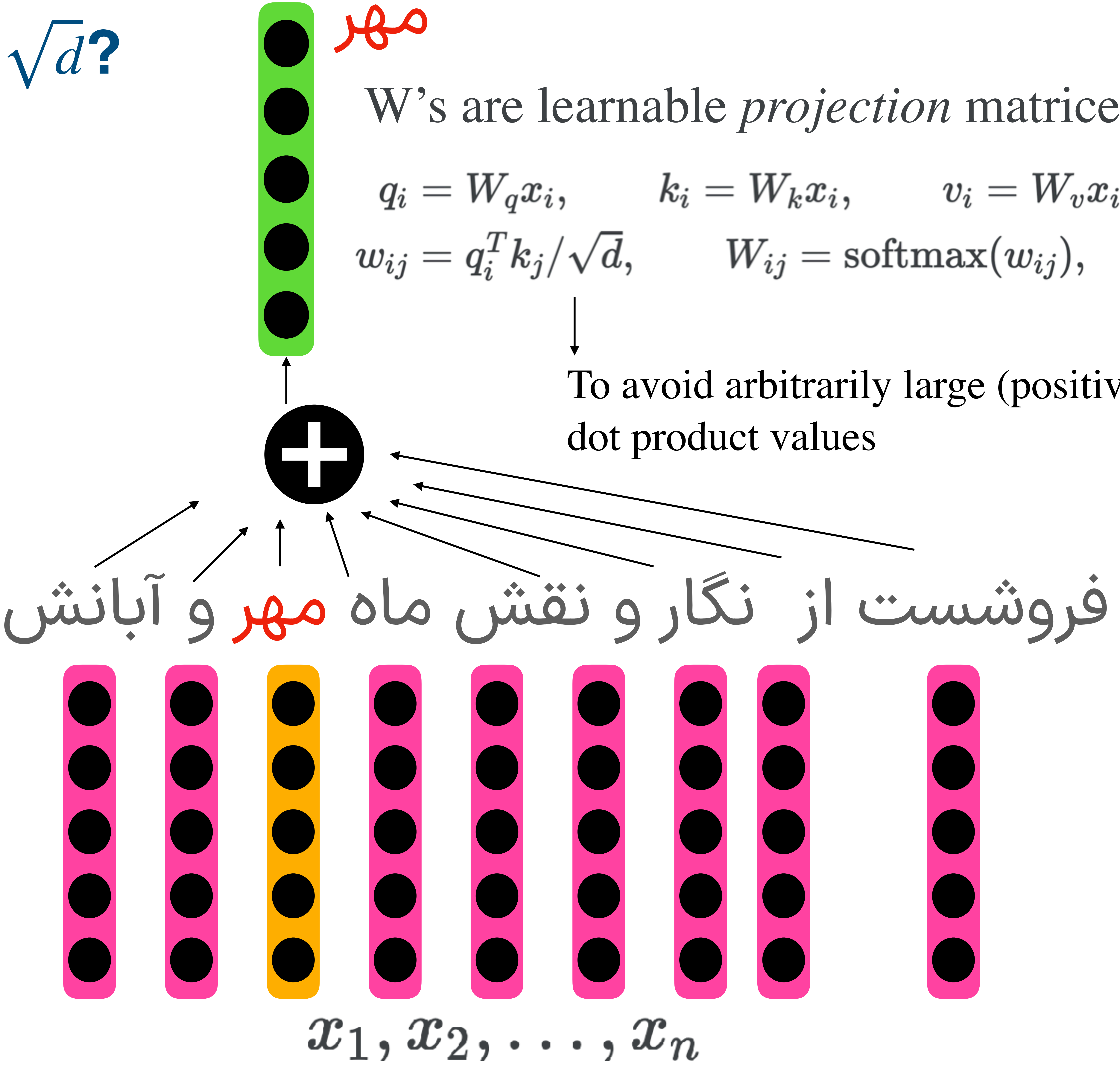
Why scaling by $\sqrt{d}$?

W's are learnable *projection* matrices

$$q_i = W_q x_i, \qquad k_i = W_k x_i, \qquad v_i = W_v x_i$$

$$w_{ij} = q_i^T k_j / \sqrt{d}, \qquad W_{ij} = \text{softmax}(w_{ij}), \qquad y_i = \sum_j W_{ij} v_j.$$

To avoid arbitrarily large (positive or negative) dot product values



WHY $\sqrt{d}$?

$$q_i = W_q x_i, \quad k_i = W_k x_i, \quad v_i = W_v x_i$$

$$w_{ij} = q_i^T k_j / \sqrt{d}, \quad W_{ij} = \mathrm{softmax}\left(w_{ij}\right), \quad y_i = \sum_j W_{ij} v_j \,.$$

Assume that **q** and **k** are unit vectors with dimension **d**, whose dimensions are independent RV with the following properties:

$$\mathrm{E}\left[q_i\right] = \mathrm{E}\left[k_i\right] = 0$$

$$\mathrm{var}\left[q_i\right] = \mathrm{var}\left[k_i\right] = 1$$

$$\begin{aligned}
\mathrm{E}[q \cdot k] &= \mathrm{E}\left[\sum_{i=1}^{d} q_i k_i\right] \\
&= \sum_{i=1}^{d} \mathrm{E}\left[q_i k_i\right] \\
&= \sum_{i=1}^{d} \mathrm{E}\left[q_i\right] \mathrm{E}\left[k_i\right] \\
&= 0
\end{aligned}$$

$$\begin{aligned}
\mathrm{var}[q \cdot k] &= \mathrm{var}\left[\sum_{i=1}^{d} q_i k_i\right] \\
&= \sum_{i=1}^{d} \mathrm{var}\left[q_i k_i\right] \\
&= \sum_{i=1}^{d} \mathrm{var}\left[q_i\right] \mathrm{var}\left[k_i\right] \\
&= \sum_{i=1}^{d} 1 \\
&= d
\end{aligned}$$

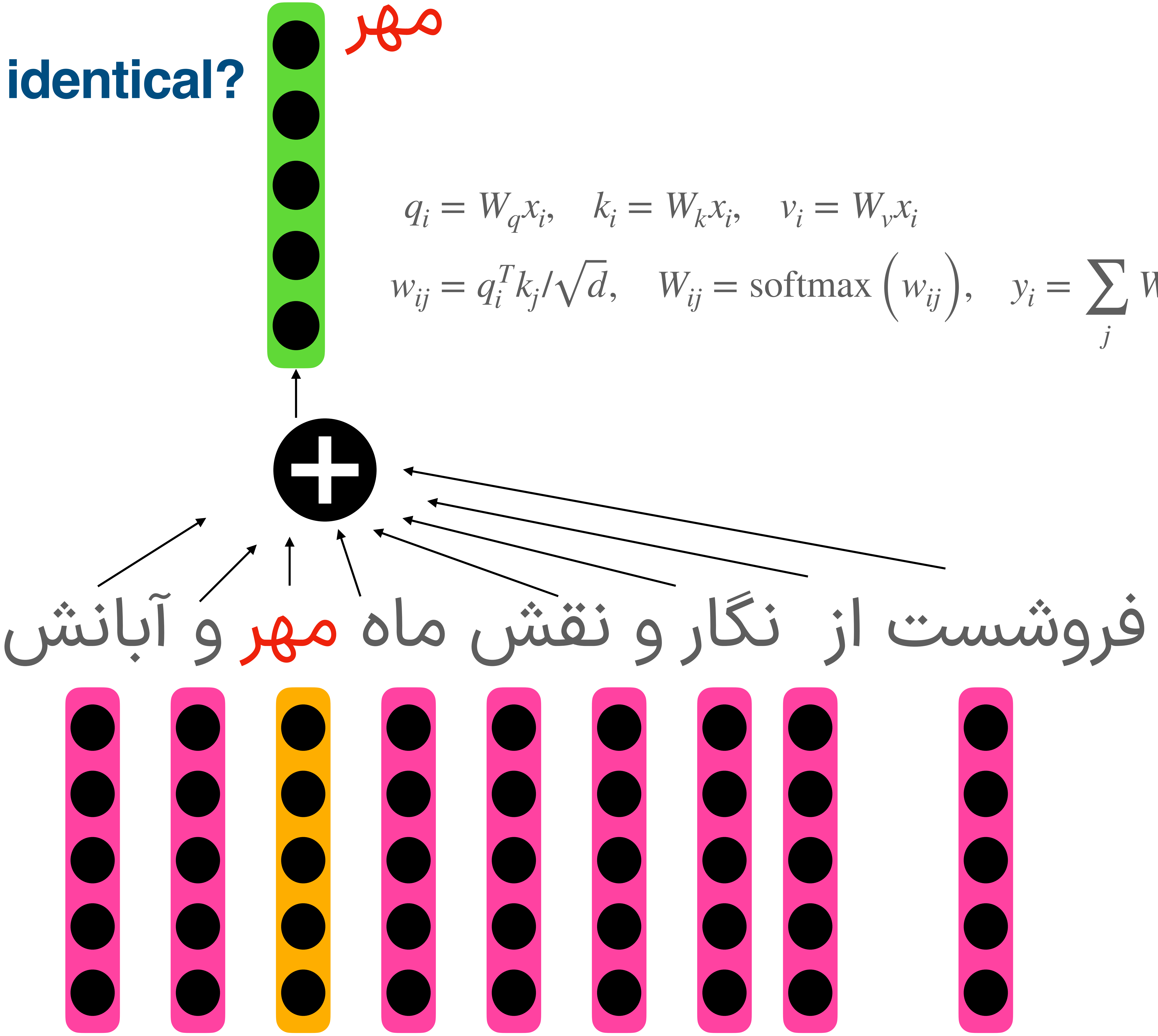
$$\longrightarrow \quad w_{ij} = \frac{q_i^T k_j - \mu}{\sigma} = \frac{q_i^T k_j}{\sqrt{d}}$$

More detailed proof: https://github.com/BAI-Yeqi/Statistical-Properties-of-Dot-Product/blob/master/proof.pdf

Are W_k and W_q identical?

$$q_i = W_q x_i, \quad k_i = W_k x_i, \quad v_i = W_v x_i$$

$$w_{ij} = q_i^T k_j / \sqrt{d}, \quad W_{ij} = \text{softmax}\left(w_{ij}\right), \quad y_i = \sum_j W_{ij} v_j \,.$$

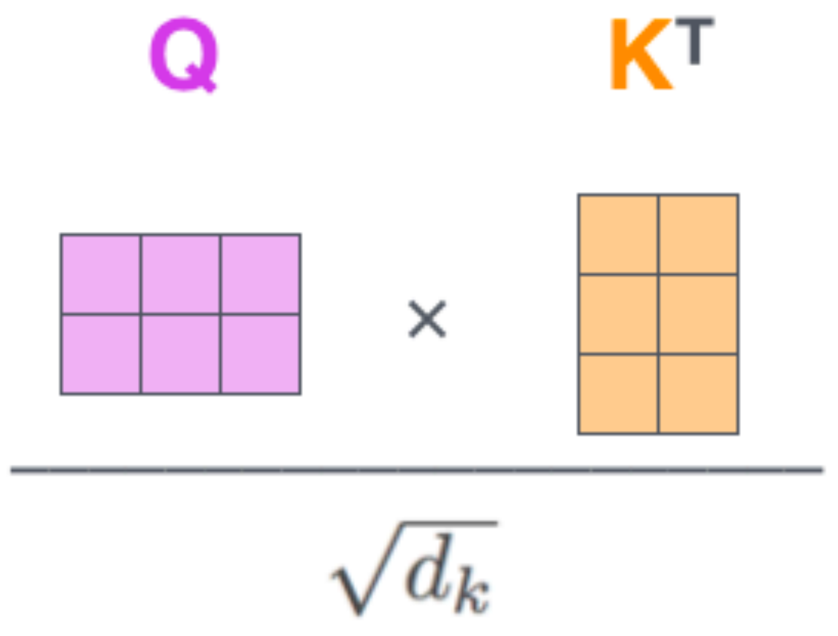


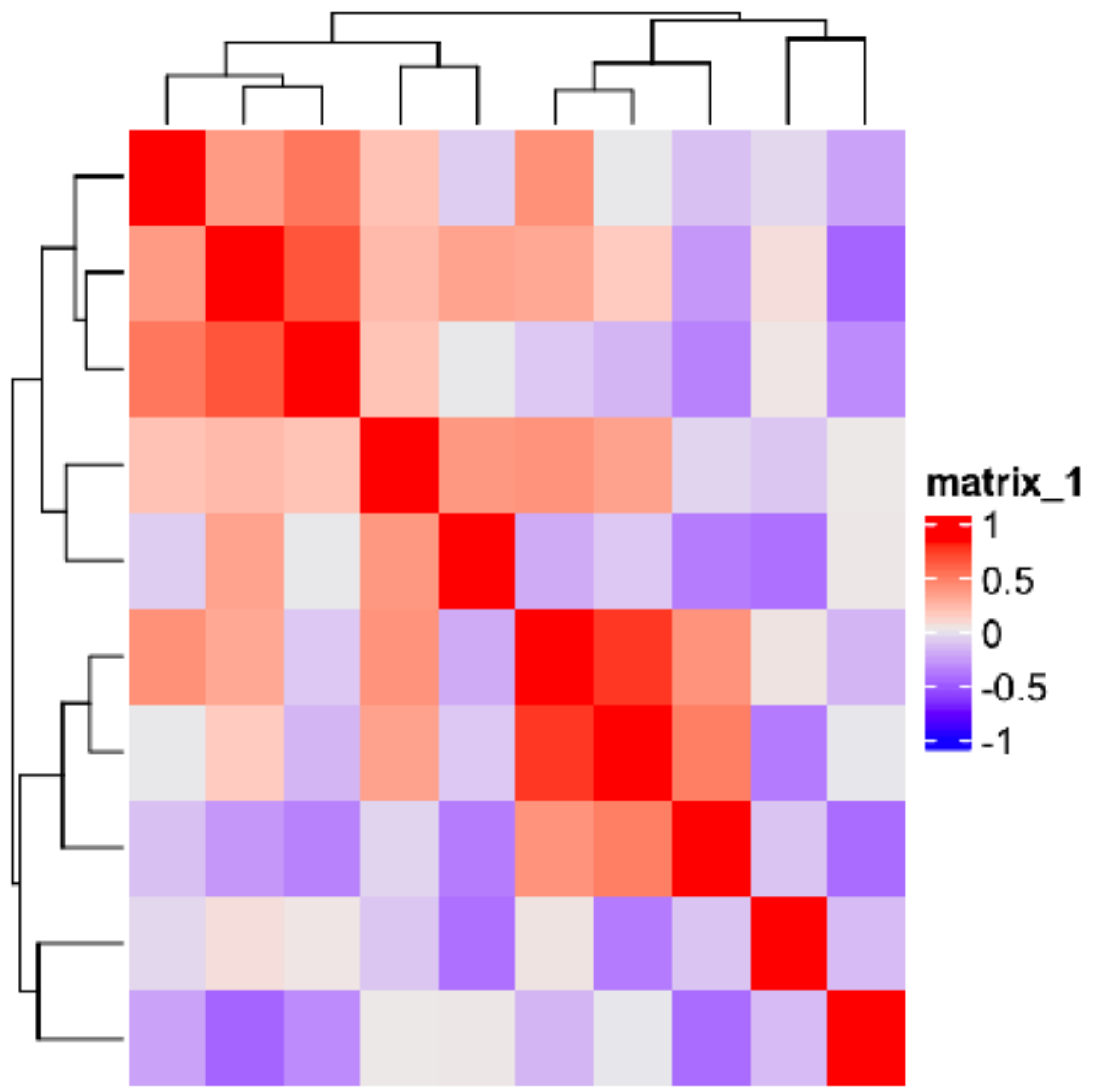
$x_1, x_2, \dots, x_n$

Are W_k and W_q identical? Better not to be identical!

$$q_i = W_q x_i, \quad k_i = W_k x_i, \quad v_i = W_v x_i$$

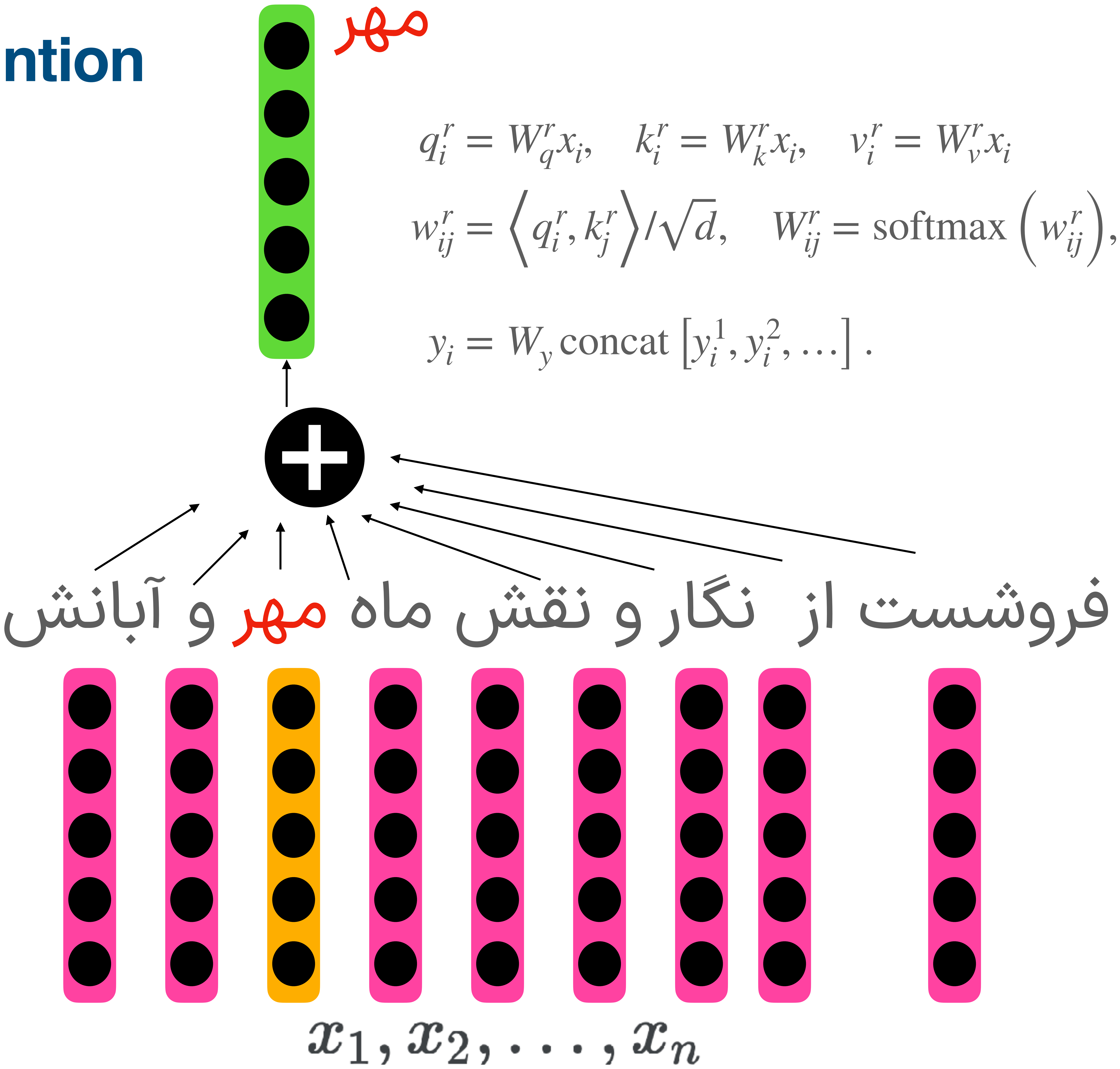
$$w_{ij} = q_i^T k_j / \sqrt{d}, \quad W_{ij} = \text{softmax}\left(w_{ij}\right), \quad y_i = \sum_j W_{ij} v_j .$$





Multihead Attention

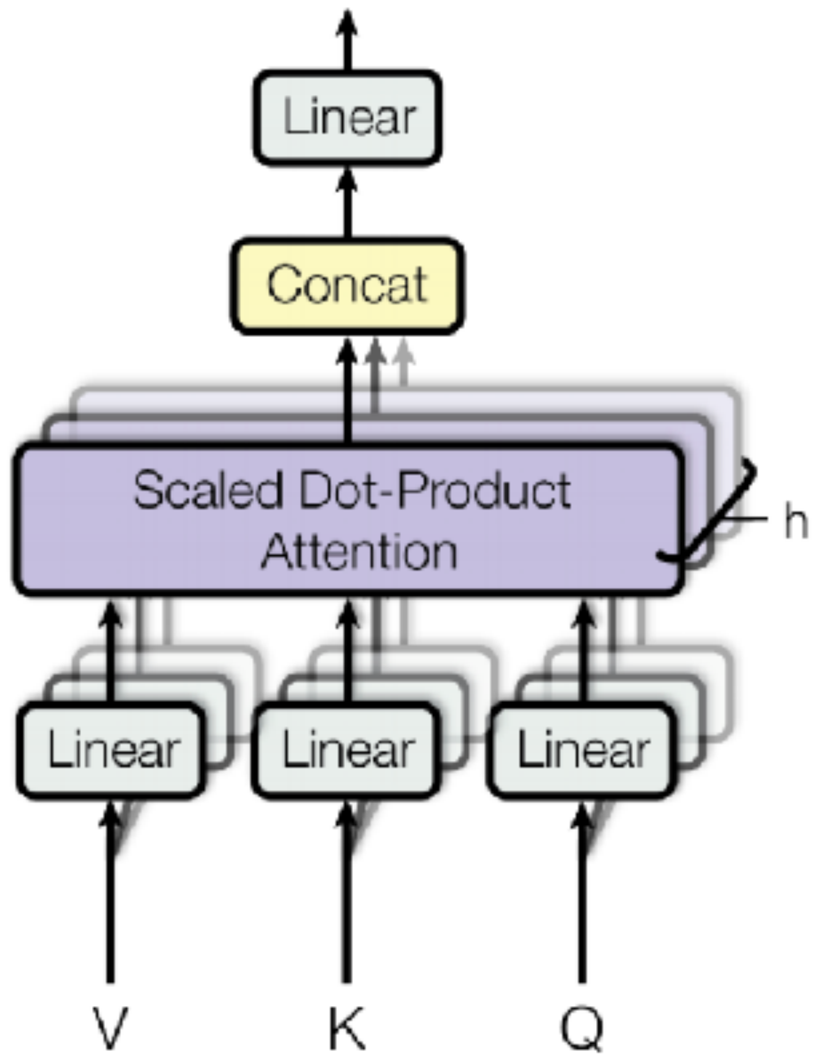
REVIEW



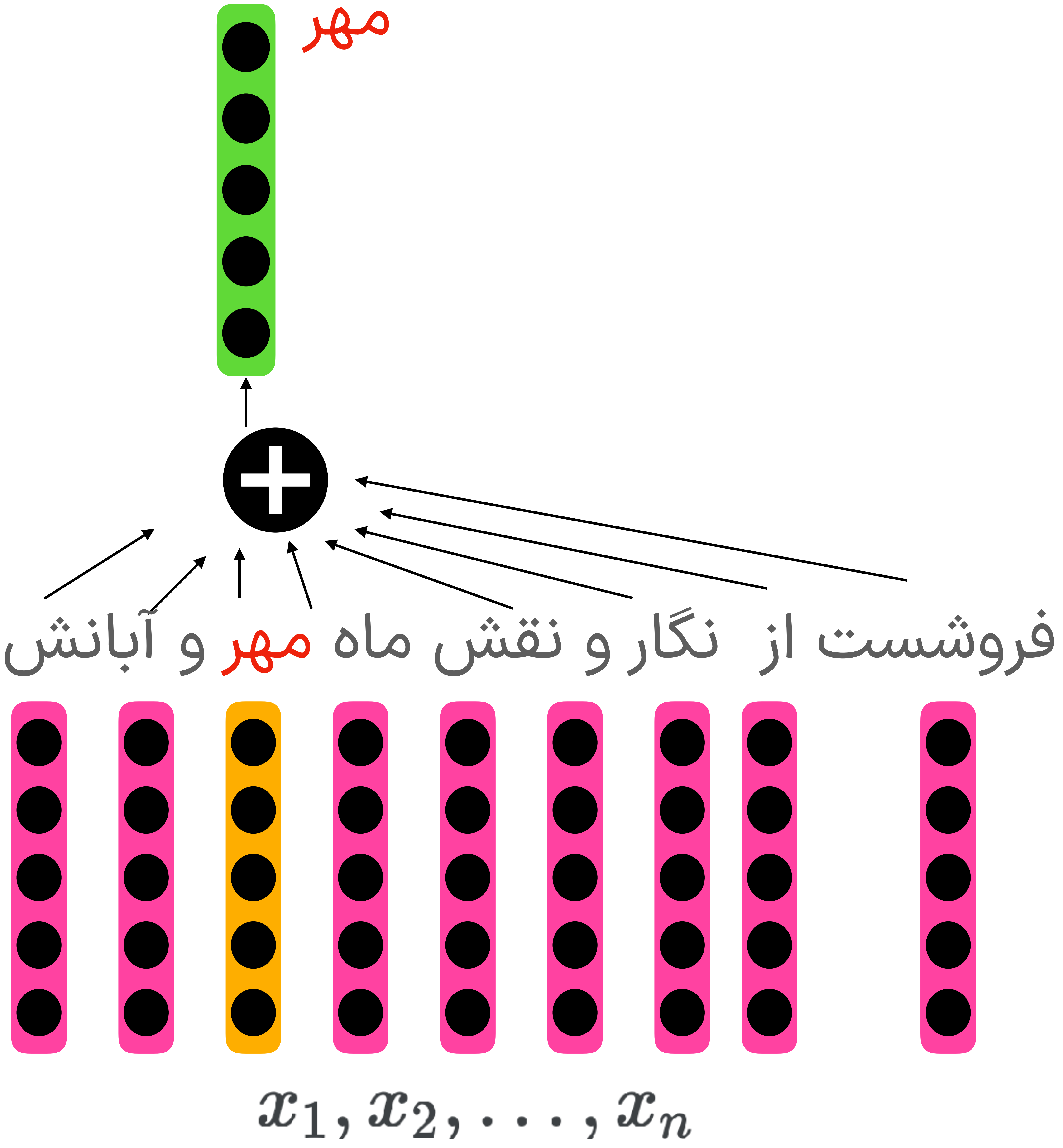
$$q_i^r = W_q^r x_i, \quad k_i^r = W_k^r x_i, \quad v_i^r = W_v^r x_i$$

$$w_{ij}^r = \left\langle q_i^r, k_j^r \right\rangle / \sqrt{d}, \quad W_{ij}^r = \mathrm{softmax}\left(w_{ij}^r\right), \quad y_i^r = \sum_j W_{ij}^r v_j,$$

$$y_i = W_y \, \mathrm{concat}\left[y_i^1, y_i^2, \dots\right].$$



Order?



Positional Embedding

- Assign a number to each time-step within the [0, 1]
  - Time-step differences are not consistent in different sentences.
- Assign a natural number to each time-step
  - Long sentences
  - Differences in the training and the inference

Positional Embedding

- ★ Unique encoding for each time-step.
- ★ Consistent distance between time-steps in varying sentence lengths.
- ★ Easily adapts to longer sentences with bounded values.
- ★ Deterministic output.

Positional Embedding types?

ABSOLUTE VS. RELATIVE POSITION ENCODING

مهر ماه باران می بارد

معمولا هر سال مهر ماه باران می بارد

Positional Embedding types?

ABSOLUTE VS. RELATIVE POSITION ENCODING

مهر ماه باران می بارد

معمولا هر سال مهر ماه باران می بارد

$$\begin{array}{cc} & \begin{array}{ccc} \text{مهر} & \text{ماه} & \text{باران} \end{array} \\ \begin{array}{c} \text{مهر} \\ \text{ماه} \\ \text{باران} \end{array} & \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \end{array} \quad \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \quad \begin{bmatrix} r_0 & r_1 & r_2 \\ r_{-1} & r_0 & r_1 \\ r_{-2} & r_{-1} & r_0 \end{bmatrix}$$

Attention Matrix *Absolute Position Bias* *Relative Position Bias*

Absolute position embeddings are favorable for classification tasks and relative embeddings perform better for span prediction tasks.

Philipp Dufter, Martin Schmitt, and Hinrich Schütze. 2022. Position Information in Transformers: An Overview. *Computational Linguistics*, 48(3):733–763.

Adding Position Embeddings

Input Embedding $U \in \mathbb{R}^{\times d}$

Position Embedding $P \in \mathbb{R}^{\times d}$

$$\tilde{\mathbf{A}} = \sqrt{\frac{1}{d}}(\mathbf{U}+\mathbf{P})\mathbf{W}^{(q)}\mathbf{W}^{(k)\intercal}(\mathbf{U}+\mathbf{P})^\intercal$$
$$\tilde{\mathbf{M}} = \mathrm{SoftMax}(\tilde{\mathbf{A}})(\mathbf{U}+\mathbf{P})\mathbf{W}^{(v)}$$
$$\tilde{\mathbf{O}} = \mathrm{LayerNorm}_2(\tilde{\mathbf{M}}+\mathbf{U}+\mathbf{P})$$
$$\tilde{\mathbf{F}} = \mathrm{ReLU}(\tilde{\mathbf{O}}\mathbf{W}^{(f_1)}+\mathbf{b}^{(f_1)})\mathbf{W}^{(f_2)}+\mathbf{b}^{(f_2)}$$
$$\tilde{\mathbf{Z}} = \mathrm{LayerNorm}_1(\tilde{\mathbf{O}}+\tilde{\mathbf{F}})$$

Philipp Dufter, Martin Schmitt, and Hinrich Schütze. 2022. Position Information in Transformers: An Overview. *Computational Linguistics*, 48(3):733–763.

Modifying Attention Matrix

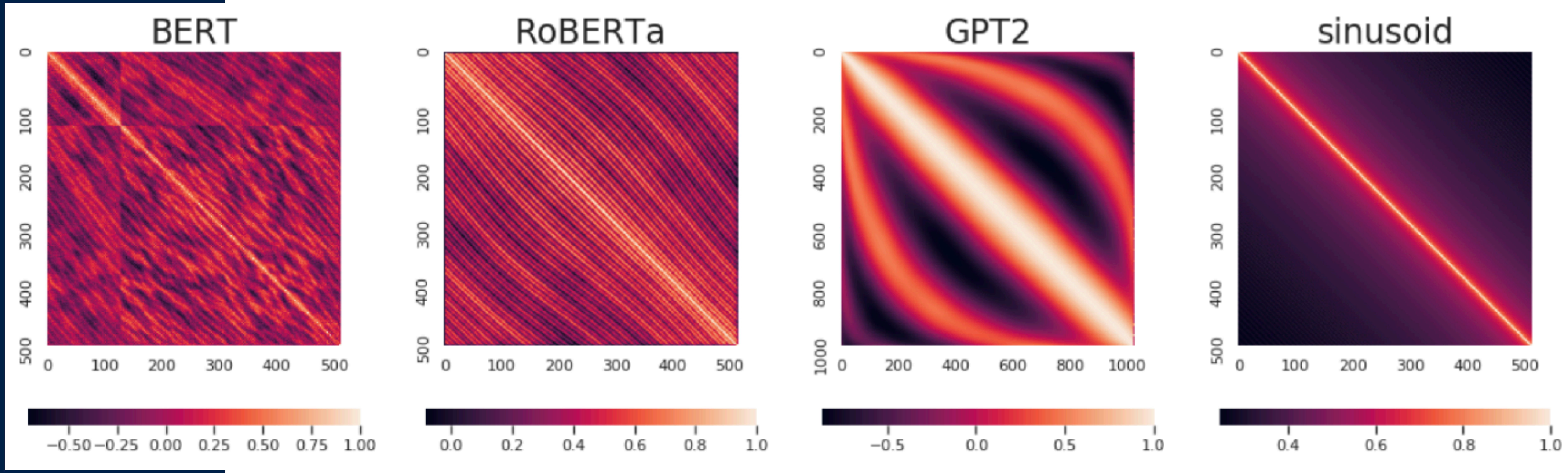
Input Embedding $U \in \mathbb{R}^{\times d}$

Position Embedding $P \in \mathbb{R}^{\times d}$

$$\hat{\mathbf{A}} \sim \underbrace{\mathbf{U}\mathbf{W}^{(q)}\mathbf{W}^{(k)\intercal}\mathbf{U}^\intercal}_{\text{unit-unit } \sim\mathbf{A}} + \underbrace{\mathbf{P}\mathbf{W}^{(q)}\mathbf{W}^{(k)\intercal}\mathbf{U}^\intercal + \mathbf{U}\mathbf{W}^{(q)}\mathbf{W}^{(k)\intercal}\mathbf{P}^\intercal}_{\text{unit-position}} + \underbrace{\mathbf{P}\mathbf{W}^{(q)}\mathbf{W}^{(k)\intercal}\mathbf{P}^\intercal}_{\text{position-position}}$$

Philipp Dufter, Martin Schmitt, and Hinrich Schütze. 2022. Position Information in Transformers: An Overview. *Computational Linguistics*, 48(3):733–763.

Positional Embedding types?



Yu-An Wang and Yun-Nung Chen. 2020. What Do Position Embeddings Learn? An Empirical Study of Pre-Trained Language Model Positional Encoding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6840–6849, Online. Association for Computational Linguistics.

Sinusoidal Positional Embedding

We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PEpos+k can be represented as a linear function of PEpos.

$$\boldsymbol{T}^{(k)}\boldsymbol{E}_{t,:} = \boldsymbol{E}_{t+k,:}$$

$$\boldsymbol{T}^{(k)} = \begin{bmatrix} \boldsymbol{\Phi}_1^{(k)} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_2^{(k)} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{\Phi}_{\frac{d_{\text{model}}}{2}}^{(k)} \end{bmatrix}$$

$$\boldsymbol{\Phi}_m^{(k)} = \begin{bmatrix} \cos(\lambda_m k) & \sin(\lambda_m k) \\ -\sin(\lambda_m k) & \cos(\lambda_m k) \end{bmatrix}$$

$$\lambda_m = 10000^{\frac{-2m}{d_{\text{model}}}}$$

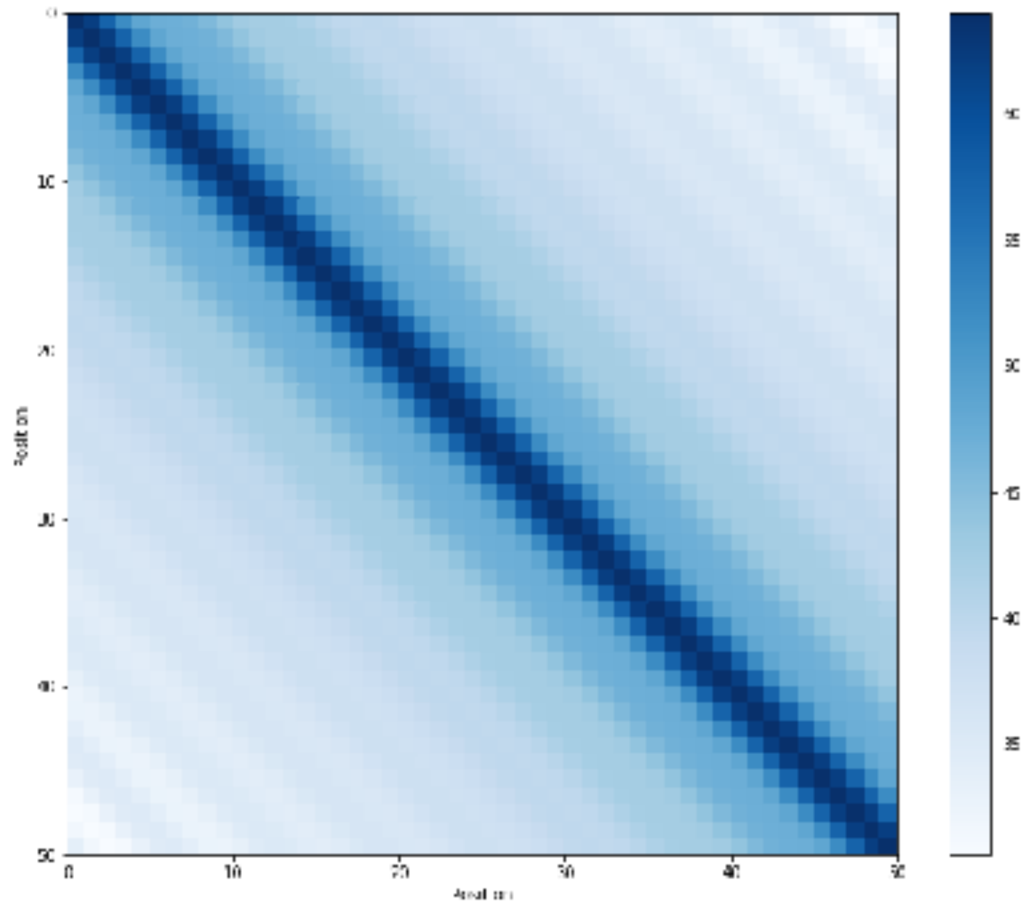
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

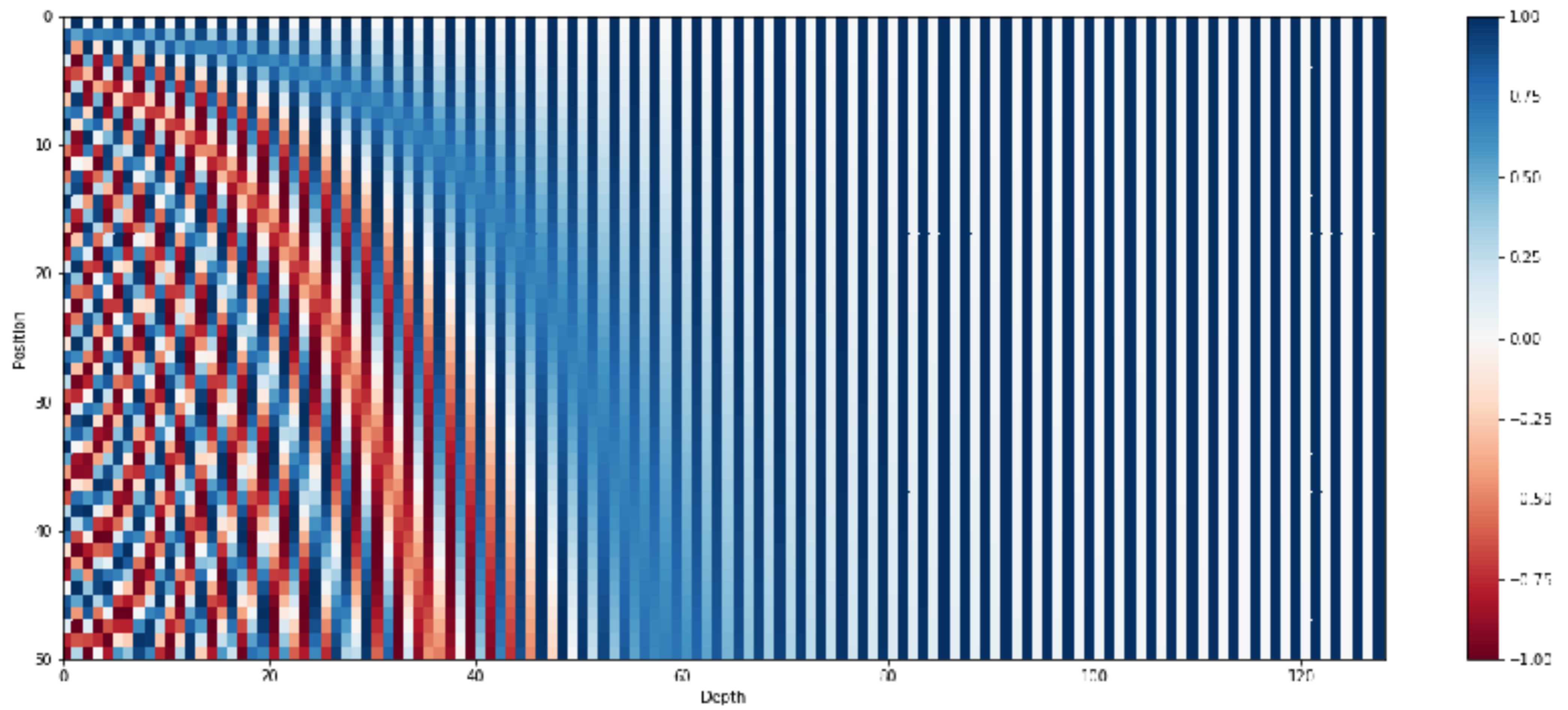
Sinusoidal Positional Embedding

$$\vec{p_t} = \begin{bmatrix} \sin(\omega_1 . t) \\ \cos(\omega_1 . t) \\ \sin(\omega_2 . t) \\ \cos(\omega_2 . t) \\ \vdots \\ \sin(\omega_{d/2} . t) \\ \cos(\omega_{d/2} . t) \end{bmatrix}_{d\times 1}$$

We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PEpos+k can be represented as a linear function of PEpos.

$$\vec{p_t}^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k . t), & \text{if } i = 2k \\ \cos(\omega_k . t), & \text{if } i = 2k+1 \end{cases} \qquad \omega_k = \frac{1}{10000^{2k/d}}$$





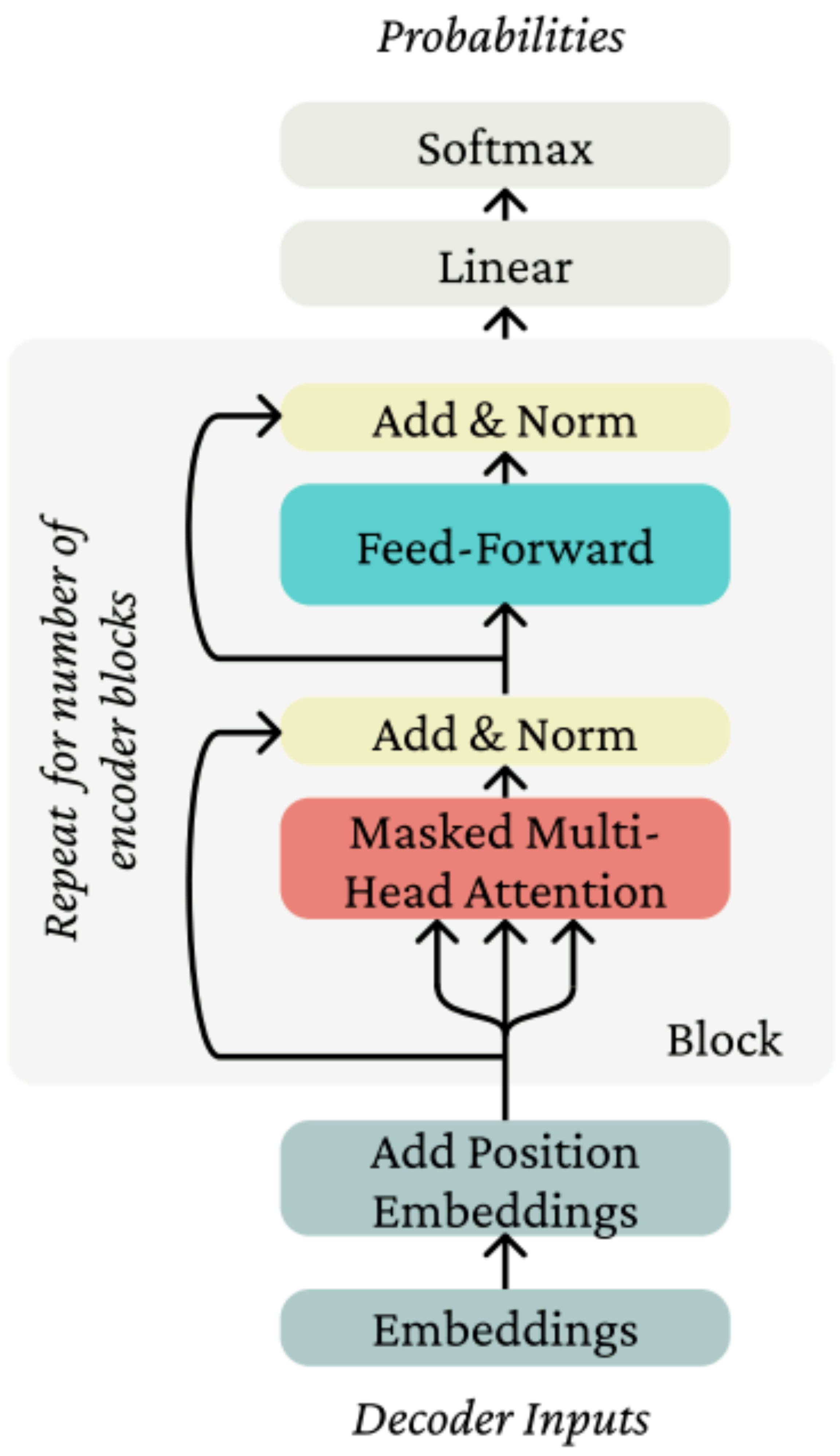
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Transformer block

- Each block has two "sublayers"
  1. Multihead attention
  2. Feed-forward NNet (with ReLU)

- Residual: x + Sublayer(x)

- Layernorm changes input to have mean 0 and variance 1



Layer normalization

Main idea: batch normalization is very helpful, but hard with sequences of different lengths

- Resulting more stable input to the next layer
- Simple solution: "layer normalization" – like batch norm, but not across the batch

Batch norm

d-dimensional vectors for each sample in batch

$a_1, a_2, \ldots, a_B$

d-dim

$$\mu = \frac{1}{B}\sum_{i=1}^{B} a_i \qquad \sigma = \sqrt{\frac{1}{B}\sum_{i=1}^{B}(a_i - \mu)^2}$$

$$\bar{a}_i = \frac{a_i - \mu}{\sigma}$$

Layer norm

different *dimensions* of a

1-dim

$$\mu = \frac{1}{d}\sum_{i=1}^{d} a_j \qquad \sigma = \sqrt{\frac{1}{d}\sum_{i=1}^{d}(a_j - \mu)^2}$$

$$\bar{a} = \frac{a - \mu}{\sigma}$$

Why transformers?

Pros:

+ Much easier to parallelize

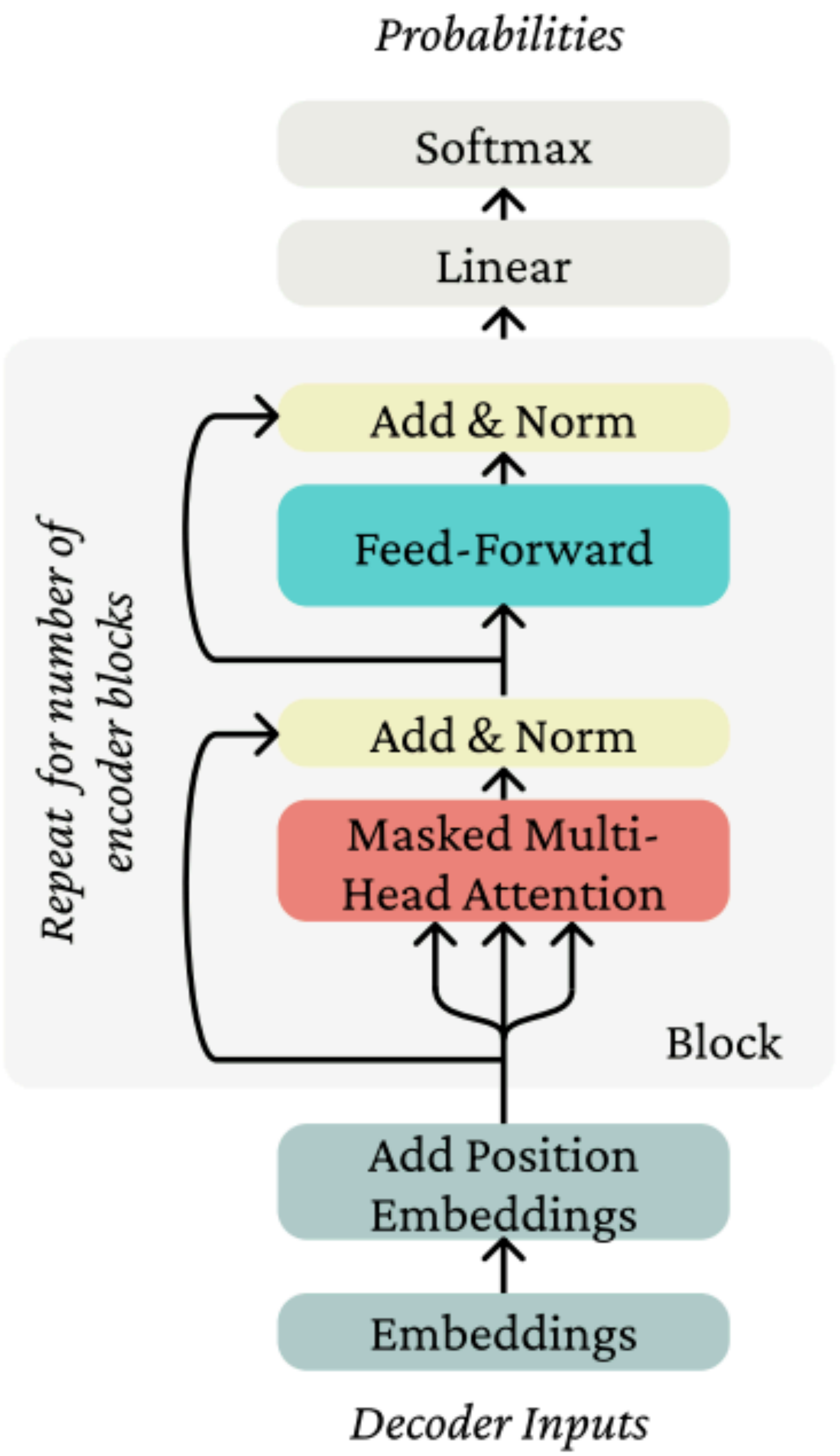
+ Much better long-range connections

+ In practice, can make it much deeper (more layers) than RNN

Cons:

- Attention computations are technically $O(n^2)$

- Somewhat more complex to implement (positional encodings, etc.)

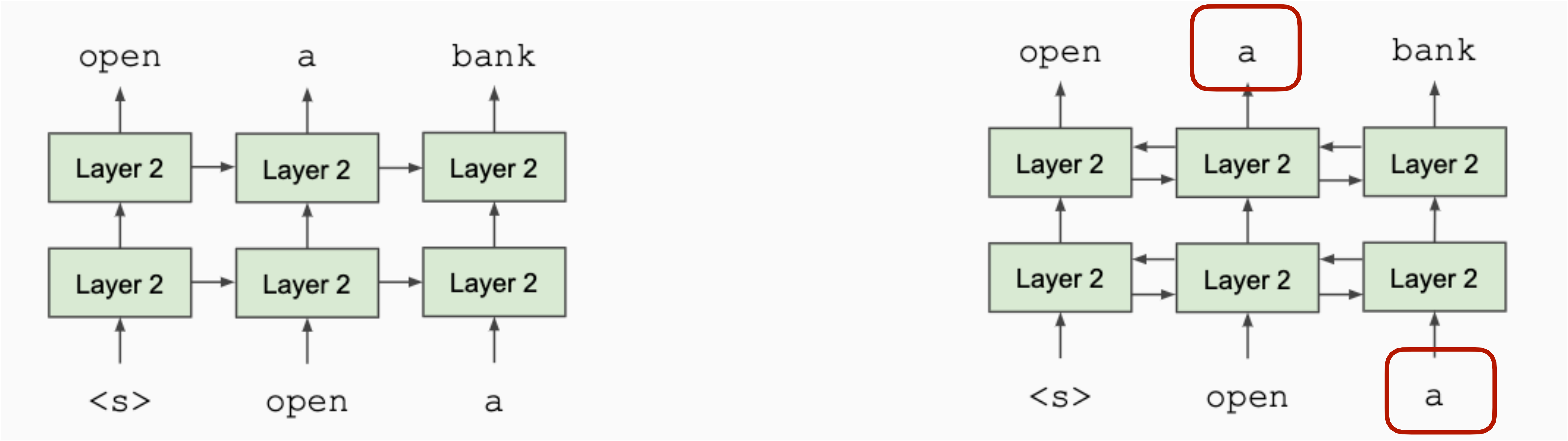


–Encoder Language Model

– BERT LM Architecture

Masked Language Modeling (MLM)

- Q: Why we can’t do language modeling with bidirectional models?

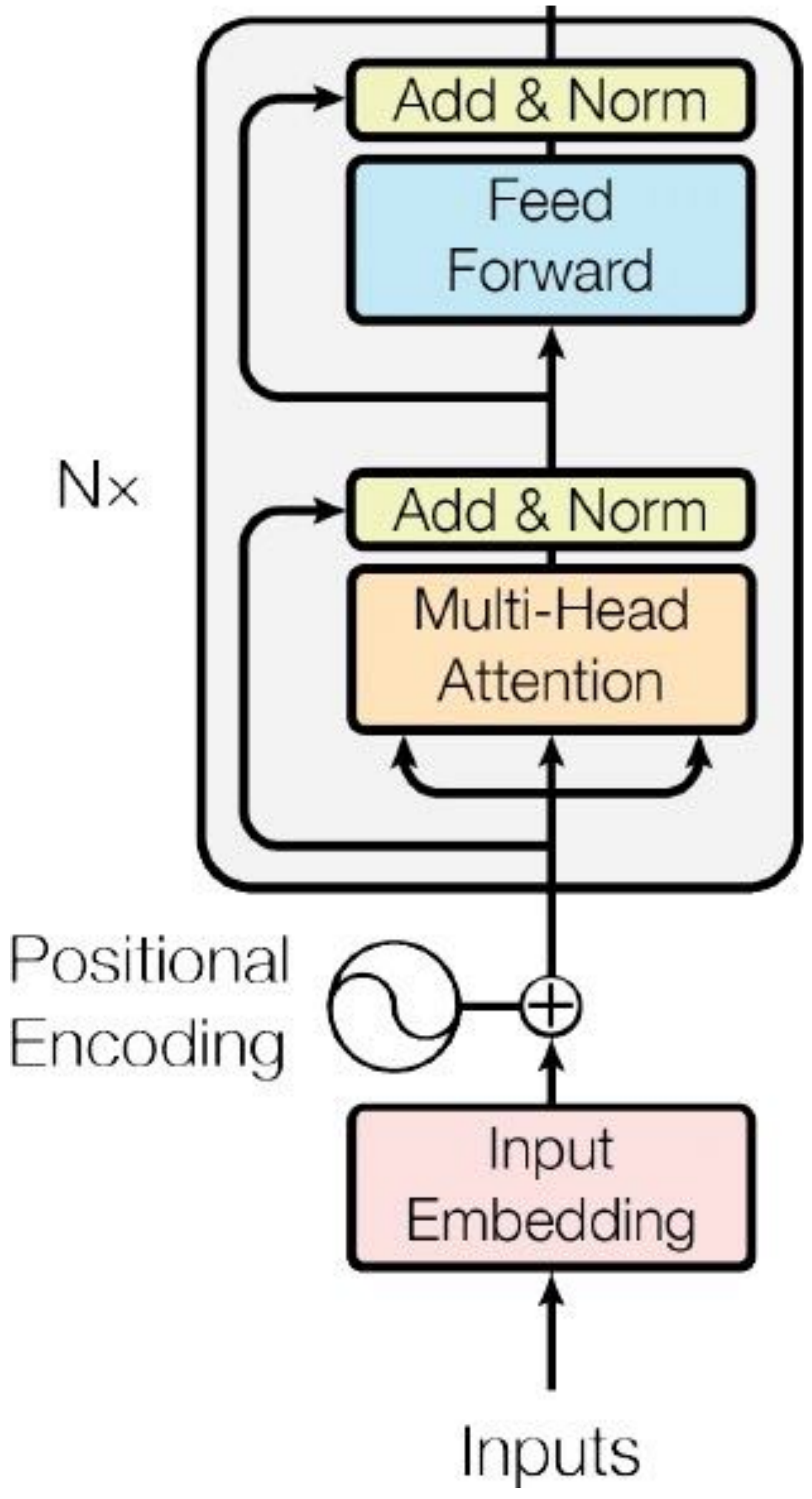


- Solution: Mask out k% of the input words, and then predict the masked words

store gallon

the man went to [MASK] to buy a [MASK] of milk

BERT pre-training: putting together



- BERT-base: 12 layers, 768 hidden size, 12 attention heads, 110M parameters

- BERT-large: 24 layers, 1024 hidden size, 16 attention heads, 340M parameters

- Training corpus: Wikipedia (2.5B) + BooksCorpus (0.8B)
- Max sequence size: 512 word pieces (roughly 256 and 256 for two non-contiguous sequences)
- Trained for 1M steps, batch size 128k

Sentence-level tasks

- Sentence pair classification tasks:

MNLI

Premise: A soccer game with multiple males playing.

Hypothesis: Some men are playing a sport.

{entailment, contradiction, neutral}

QQP

Q1: Where can I learn to invest in stocks?

Q2: How can I learn more about stocks?

{duplicate, not duplicate}

- Single sentence classification tasks:

SST2

rich veins of funny stuff in this movie

{positive, negative}

Token-level tasks

- Extractive question answering e.g., SQuAD (Rajpurkar et al., 2016)

SQuAD

Question: The New York Giants and the New York Jets play at which stadium in NYC ?

Context: The city is represented in the National Football League by the New York Giants and the New York Jets , although both teams play their home games at MetLife Stadium in nearby East Rutherford , New Jersey , which hosted Super Bowl XLVIII in 2014 .

(Training example 29,883)

MetLife Stadium

- Named entity recognition (Tjong Kim Sang and De Meulder, 2003)

CoNLL 2003 NER

John	Smith	lives	in	New	York
B-PER	I-PER	O	O	B-LOC	I-LOC

بسم الله الرحمن الرحیم

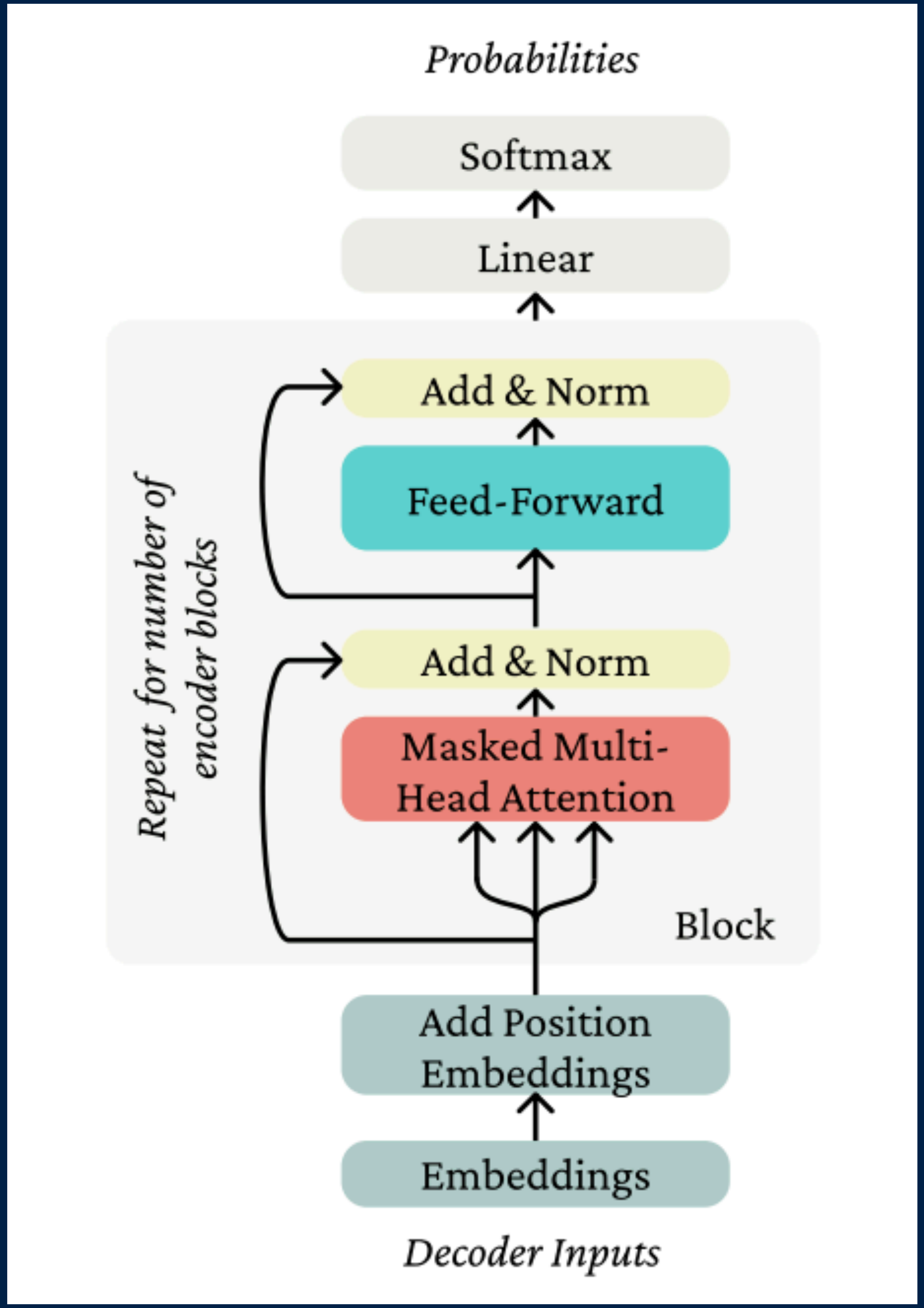
Large Language models

Lecture 4 - Transformers (iii)

Oct. 14th 2023

Artificial Intelligence Group
Computer Engineering Department, SUT

–Transformer Block



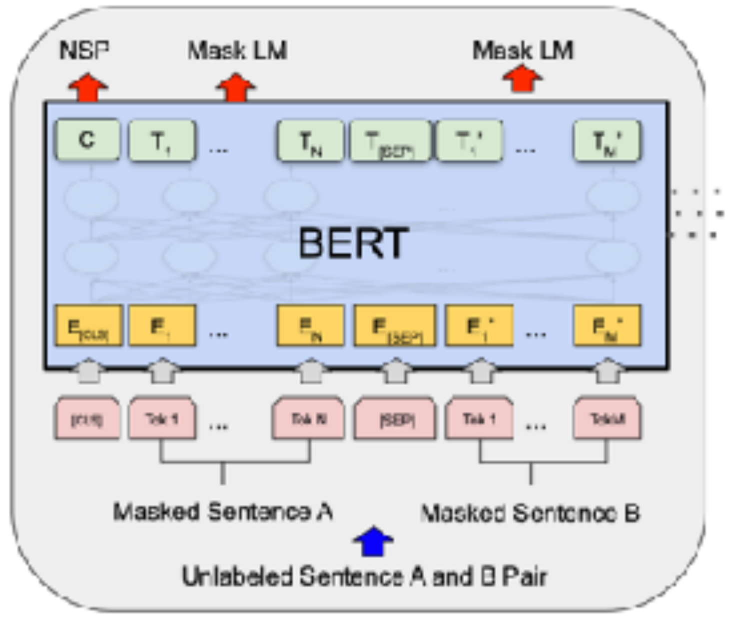
−Encoder Language Model

Encoder LM

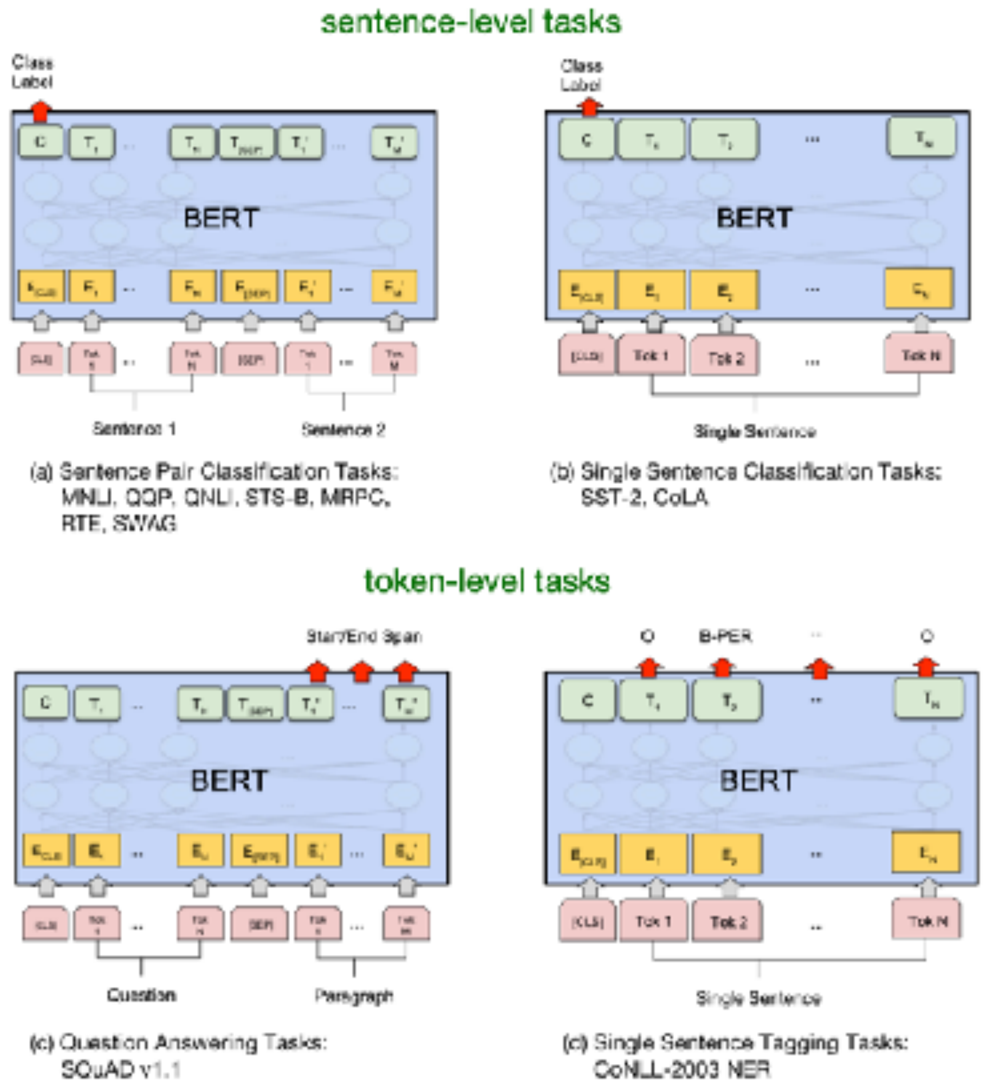
- BERT
- Variations

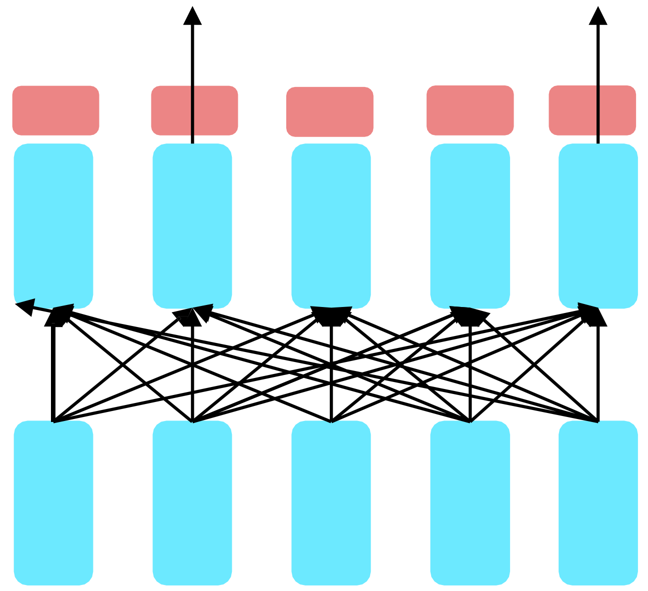
Encoder Language Model

$$P(x) = \prod_{i=1}^{n} P(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$



Pre-training





$$h_1, \ldots, h_t = \text{Decoder}(\mathrm{x}_1, \ldots, \mathrm{x}_t)$$

$$\mathrm{x}_{\text{mask}} \sim A\mathrm{h}_{\text{masked}} + b$$

BERT: key contributions

- It is a **fine-tuning approach** based on a deep **Transformer encoder**
- The key: learn representations based on **bidirectional context**

Why? Because both left and right contexts are important to understand the meaning of words.

Example #1: we went to the river bank.

Example #2: I need to go to bank to make a deposit.

- **Pre-training objectives**: masked language modeling + next sentence prediction
- State-of-the-art performance on a large set of **sentence-level** and **token-level** tasks

MLM: masking rate and strategy

- **Q: What is the value of *k*?**
  - They always use k = 15%.
  - Too little masking: computationally expensive (we need to increase # of epochs)
  - Too much masking: not enough context
  - See (Wettig et al., 2022) for more discussion of masking rates

- **Q: How are masked tokens selected?**
  - 15% tokens are uniformly sampled
  - Is it optimal? See span masking (Joshi et al., 2020) and PMI masking (Levine et al., 2021)

Example: He [MASK] from Kuala [MASK] , Malaysia.

Next Sentence Prediction (NSP)

- Motivation: many NLP downstream tasks require understanding the relationship between two sentences (natural language inference, paraphrase detection, QA)
- NSP is designed to reduce the gap between pre-training and fine-tuning

[CLS]: a special token always at the beginning

[SEP]: a special token used to separate two segments

Input = [CLS] the man went to [MASK] store [SEP]

he bought a gallon [MASK] milk [SEP]

Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]

penguin [MASK] are flight ##less birds [SEP]

Label = NotNext

They sample two contiguous segments for 50% of the time and another random segment from the corpus for 50% of the time

BERT Training

Dataset. Let $\mathcal{D}$ be a set of examples $(x_{1:L}, c)$ constructed as follows:

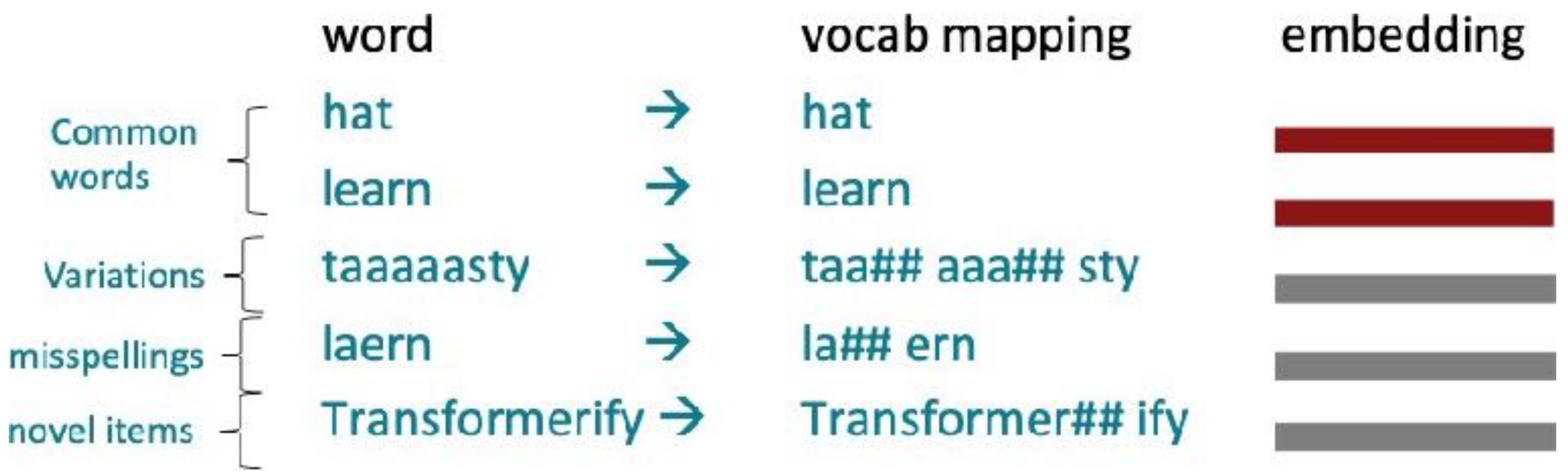
- Let A be a sentence from the corpus.
- With probability 0.5, let B be the next sentence.
- With probability 0.5, let B be a random sentence from the corpus.
- Let $x_{1:L} = [\text{[CLS]}, A, \text{[SEP]}, B]$.
- Let c denote whether B is the next sentence or not.

Objective. Then the BERT objective is:

$$\mathcal{O}(\theta) = \sum_{(x_{1:L},c)\in\mathcal{D}} \underbrace{\mathbb{E}_{I,\tilde{x}_{1:L}\sim A(\cdot\mid x_{1:L},I)}\left[\sum_{i\in I} -\log p_\theta(\tilde{x}_i \mid x_{1:L})\right]}_{\text{masked language modeling}} + \underbrace{-\log p(c \mid \phi(x_{1:L})_1)}_{\text{next sentence prediction}}.$$

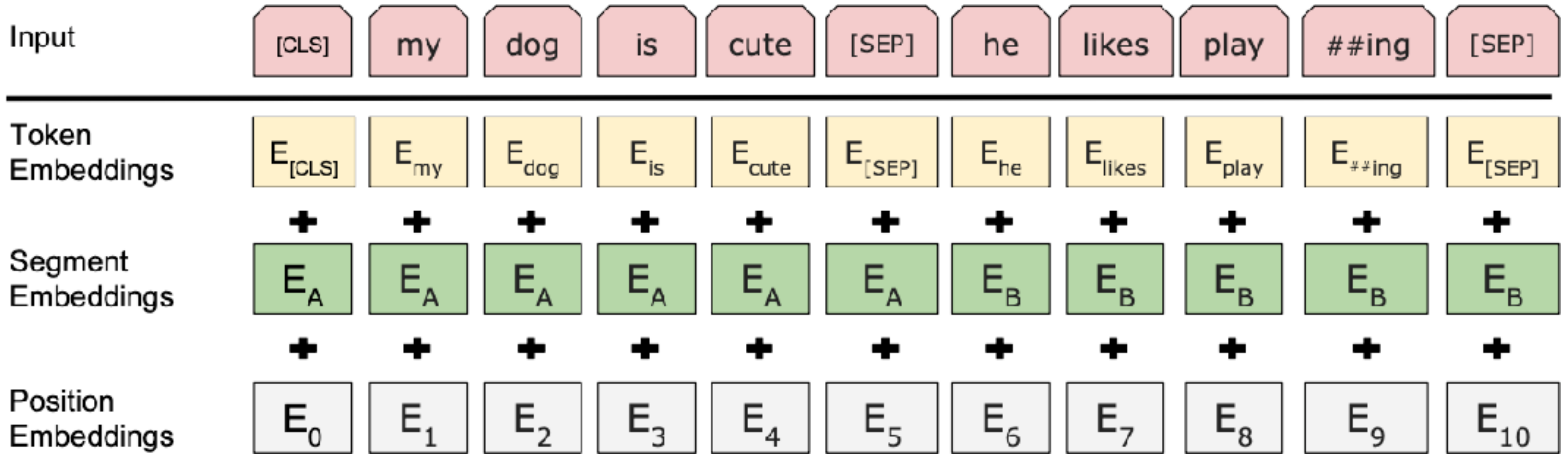
BERT pre-training: putting together

- Vocabulary size: 30,000 wordpieces (common sub-word units) (Wu et al., 2016)



(Image: Stanford CS224N)

- Input embeddings:



Separate two segments

- Just two possible "segment embeddings": EA and EB.
- Positional embeddings are learned vectors for every possible position between 0 and 512-1.

Byte Pair Encoding (BPE)

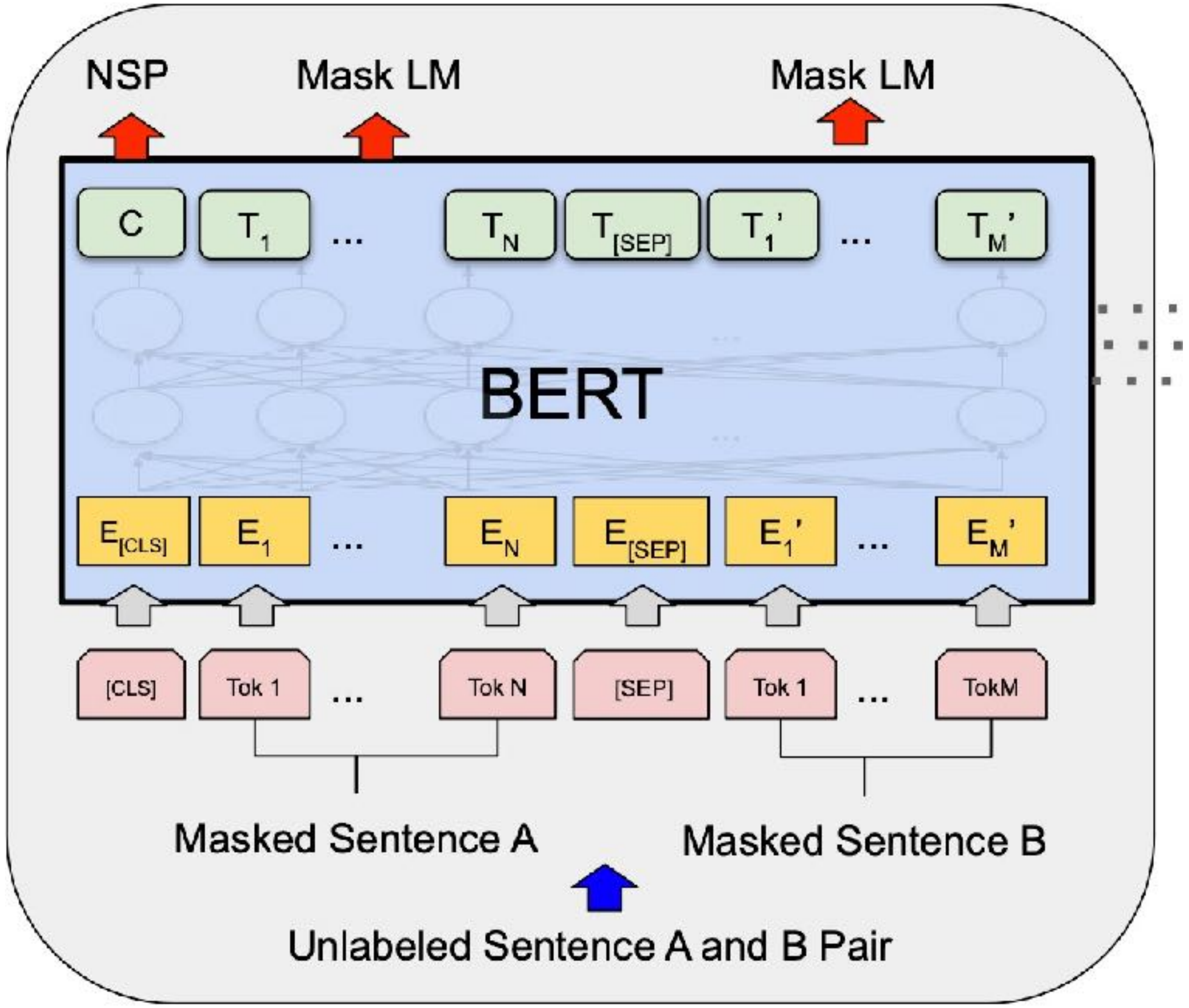
- Step 0: Set up vocabulary.
- Step 1: Represent words using characters + end token </w>.
- Step 2: Count character pairs in vocabulary.
- Step 3: Merge highest frequency pairs, add new n-gram.
- Step 4: Continue merging until reaching desired vocab size or merge count.

Unicode: we can run BPE on bytes instead of Unicode characters (Wang et al. 2019).

As we have **(144,697) of Unicode characters.**

```
u-n-r-e-l-a-t-e-d
u-n re-l-a-t-e-d
u-n re-l-at-e-d
u-n re-l-at-ed
un re-l-at-ed
un re-l-ated
un rel-ated
un-related
unrelated
```

BERT pre-training: putting together



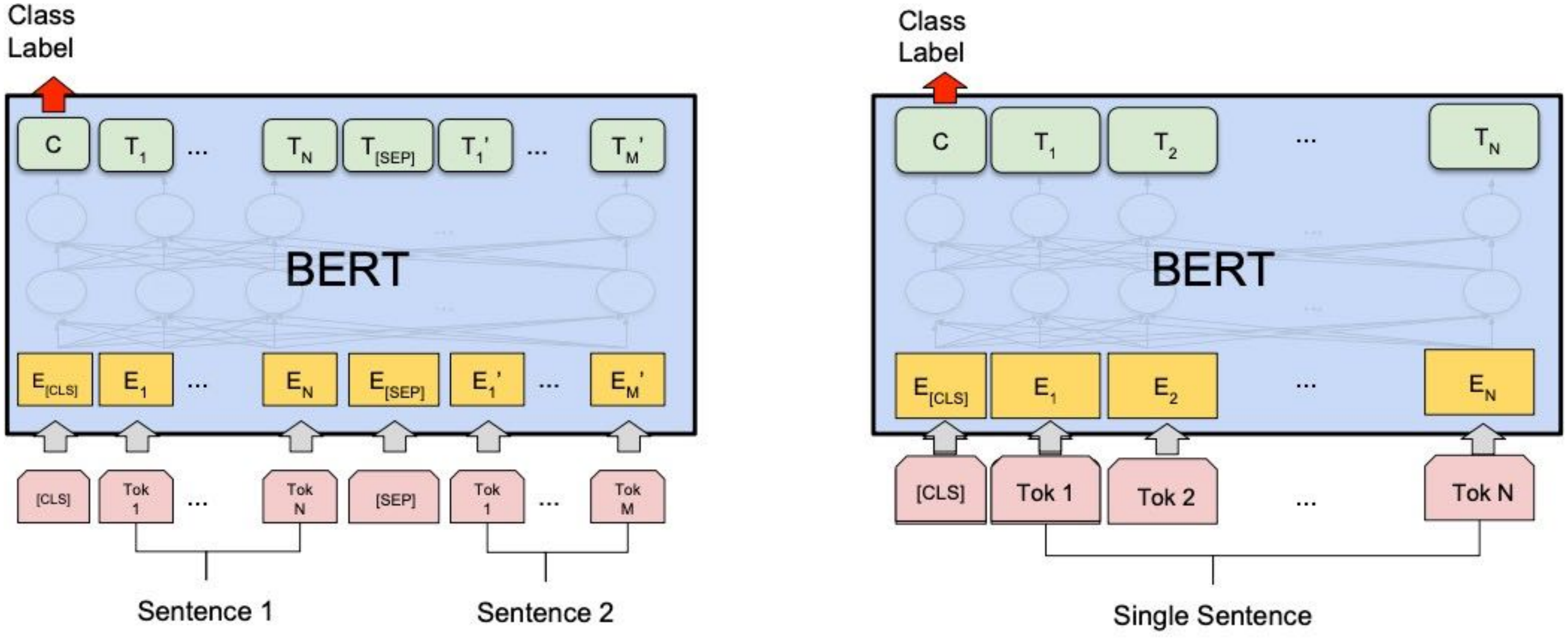
Pre-training

- MLM and NSP are trained together
- [CLS] is pre-trained for NSP
- Other token representations are trained for MLM

Fine-tuning BERT

"Pretrain once, finetune many times."

sentence-level tasks



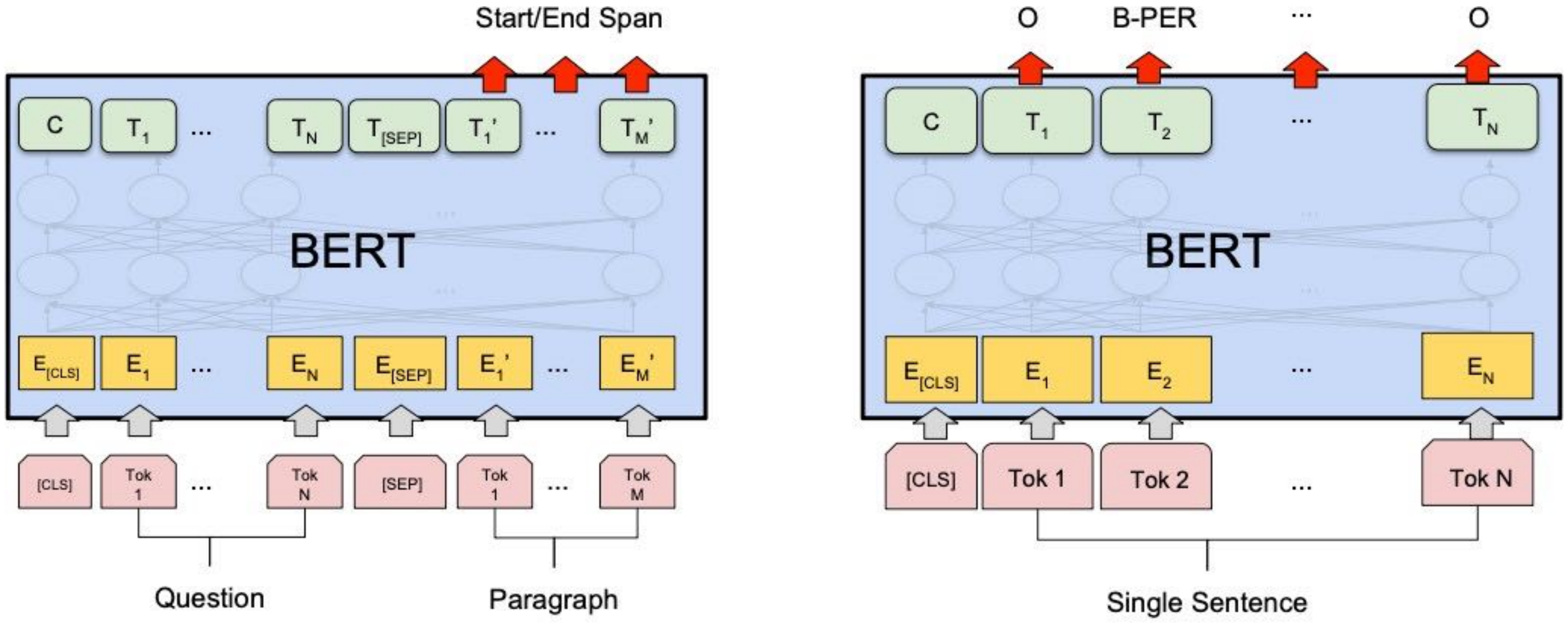
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

Fine-tuning BERT

“Pretrain once, finetune many times.”

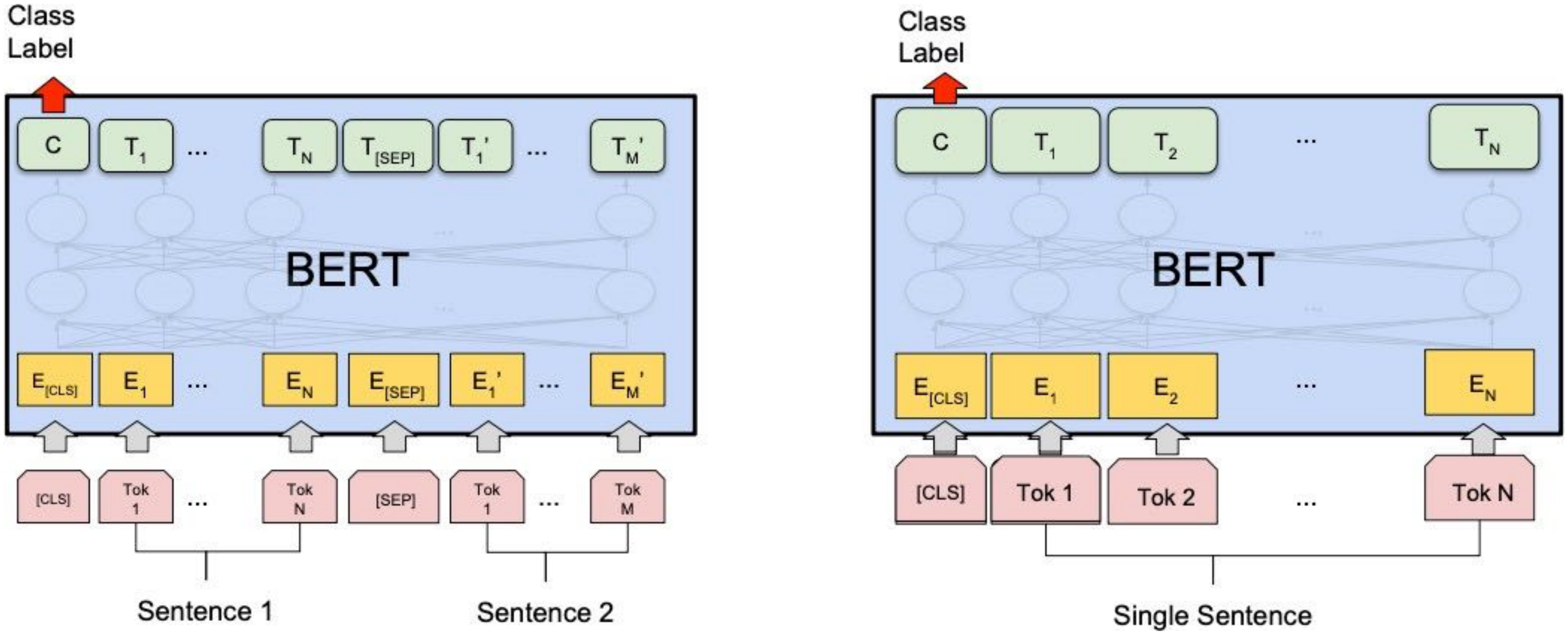
token-level tasks



(c) Question Answering Tasks:
SQuAD v1.1

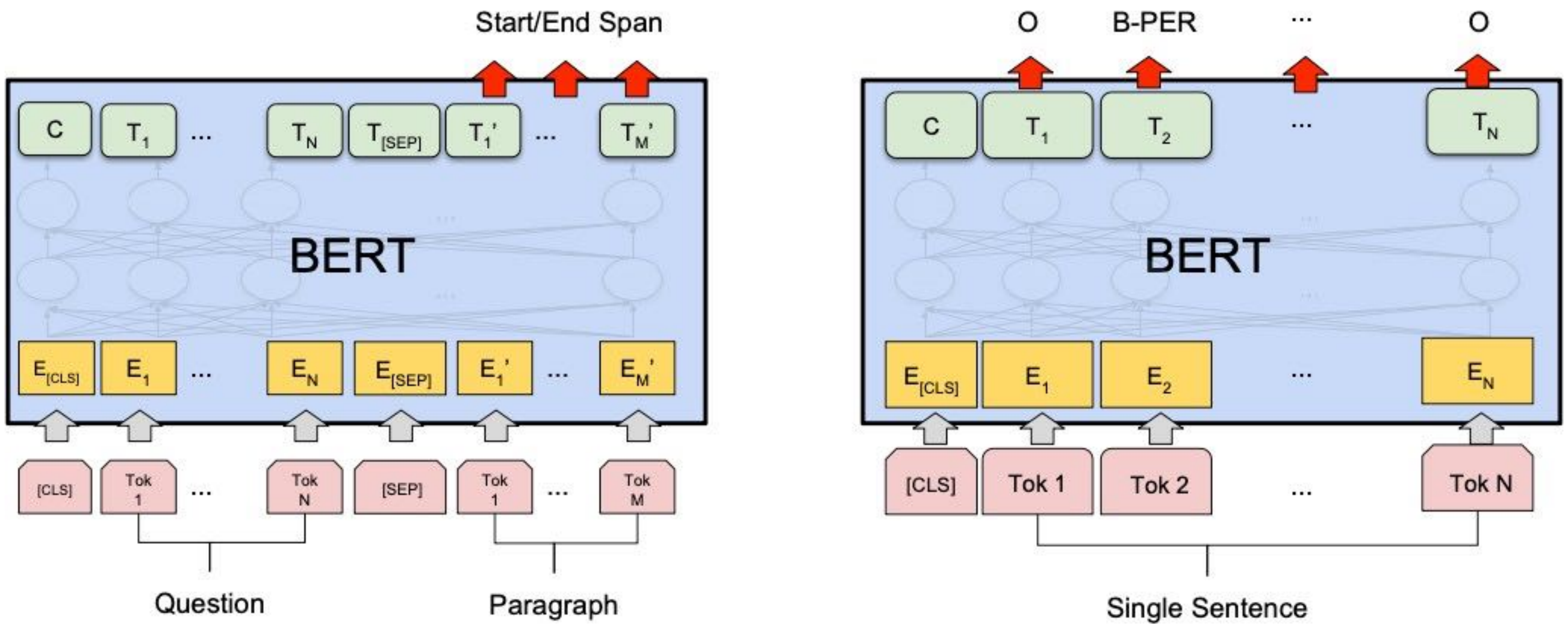
(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

Fine-tuning BERT



- For sentence pair tasks, use [SEP] to separate the two segments with segment embeddings
- Add a linear classifier on top of [CLS] representation

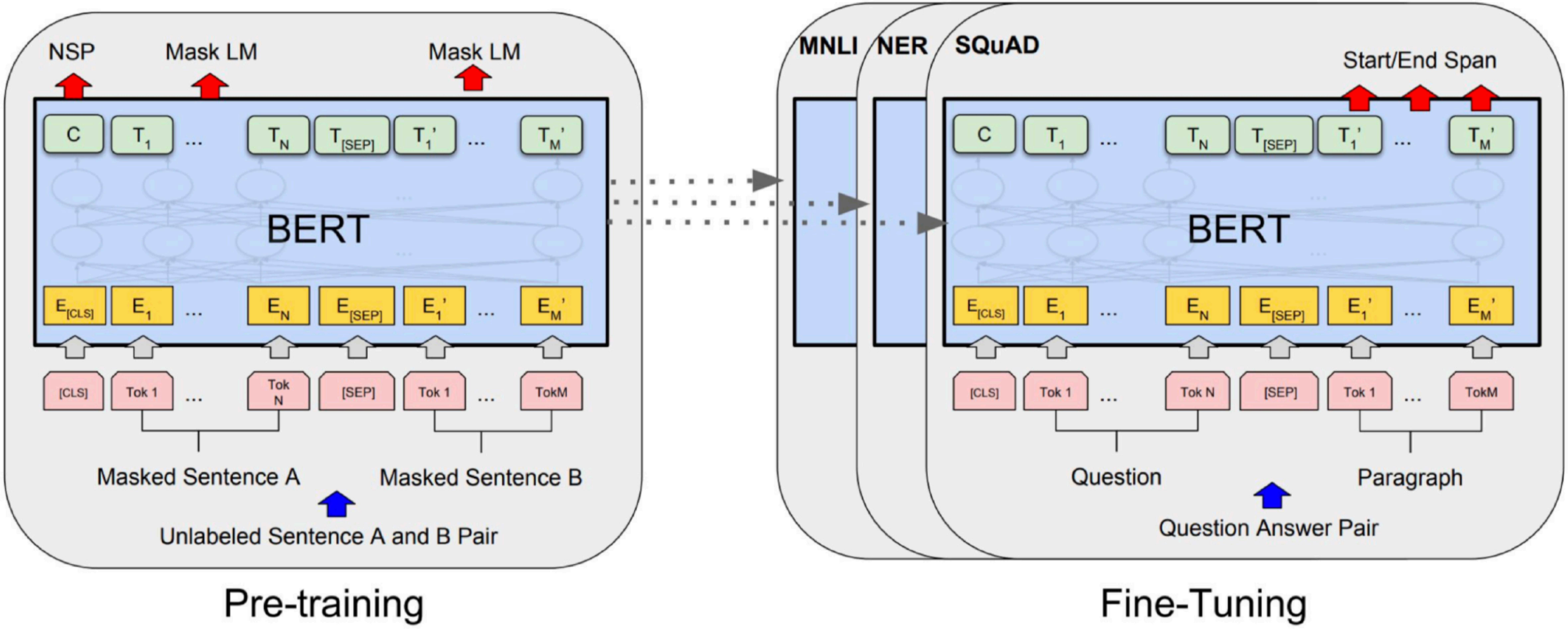
Fine-tuning BERT



- For token-level prediction tasks, add linear classifier on top of hidden representations

Q: How many new parameters?

Finetuning Paradigm in NLP



BERT Extensions

- Models that handle long contexts ($\gg$ 512 tokens)
  - Longformer, Big Bird, …
- Multilingual BERT
  - Trained single model on 104 languages from Wikipedia. Shared 110k WordPiece vocabulary
- BERT extended to different domains
  - SciBERT, BioBERT, FinBERT, ClinicalBERT, …
- Making BERT smaller to use
  - DistillBERT, TinyBERT, …

BERT Extensions

- RoBERTa (Liu et al., 2019)
  - Trained on 10x data & longer, no NSP
  - Much stronger performance than BERT (e.g., 94.6 vs 90.9 on SQuAD)
  - Still one of the most popular models to date
- ALBERT (Lan et al., 2020)
  - Increasing model sizes by sharing model parameters across layers
  - Less storage, much stronger performance but runs slower..

What happened after BERT?

Lots of people are trying to understand what BERT has learned and how it works

A Primer in BERTology: What We Know About How BERT Works

Anna Rogers
Center for Social Data Science
University of Copenhagen
arogers@sodas.ku.dk

Olga Kovaleva
Dept. of Computer Science
University of Massachusetts Lowell
okovalev@cs.uml.edu

Anna Rumshisky
Dept. of Computer Science
University of Massachusetts Lowell
arum@cs.uml.edu

- Syntactic knowledge, semantic knowledge, world knowledge…
- How to mask, what to mask, where to mask, alternatives to masking..

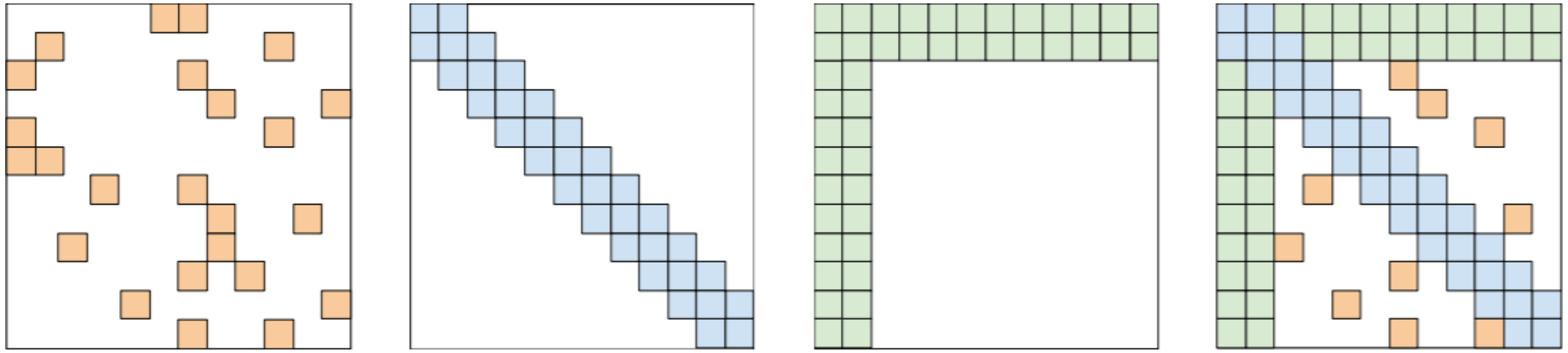
Encoder LM

- BERT
- Variations

Reducing Attention Cost

- **BigBird** [Zaheer et al., 2021]

Key idea: replace all-pairs interactions with a family of other interactions, **like local windows**, **looking at everything**, and **random interactions**.



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

–Decoder Language Model

– GPT LM Architecture

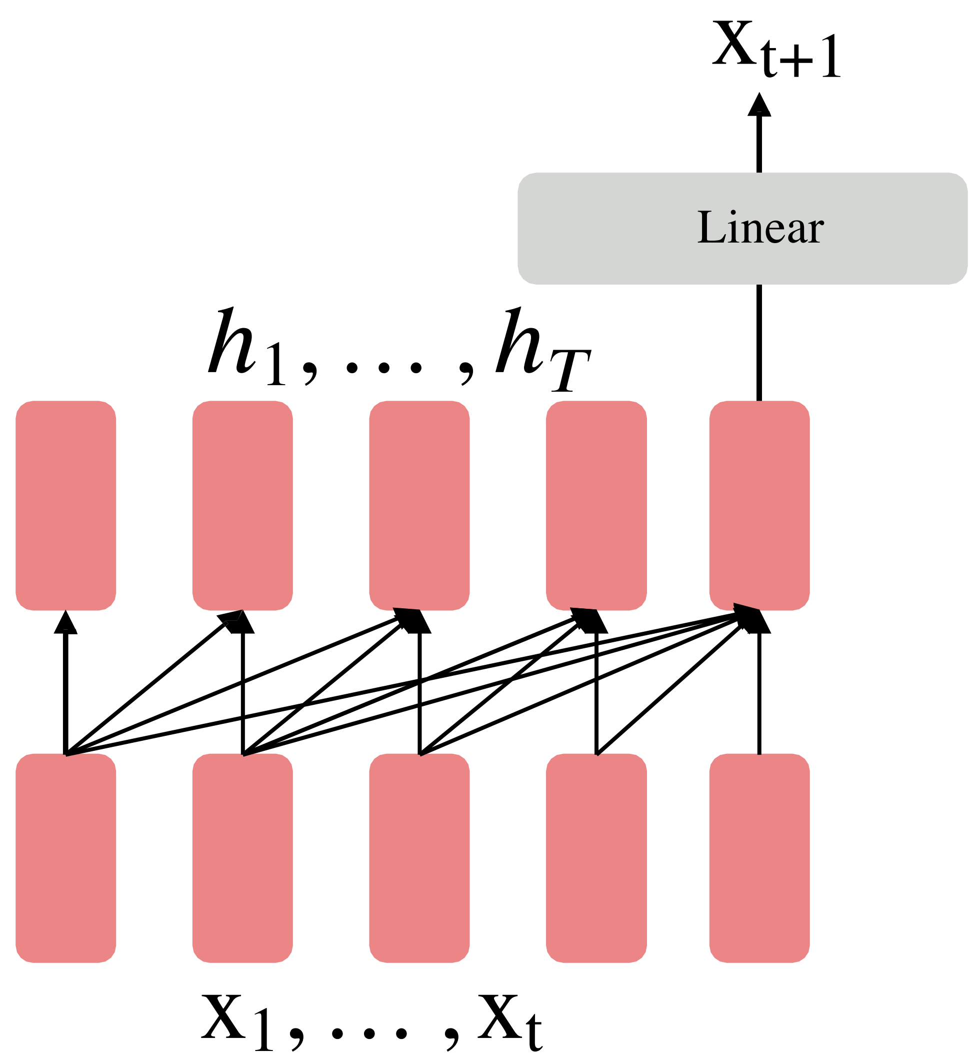
Decoder

- **GPT-models**

Decoder Language Model

Autoregressive (AR) models use decoder stacks in generation, aiming to maximize log-likelihood via forward autoregressive factorization:

$$\max_{\theta} \log p_\theta(x_1, \ldots, x_T) \approx \sum_{t=1}^{T} \log p_\theta(x_t \mid x_1, \ldots, x_{t-1})$$



$$h_1, \ldots, h_t = \text{Decoder}(x_1, \ldots, x_t)$$

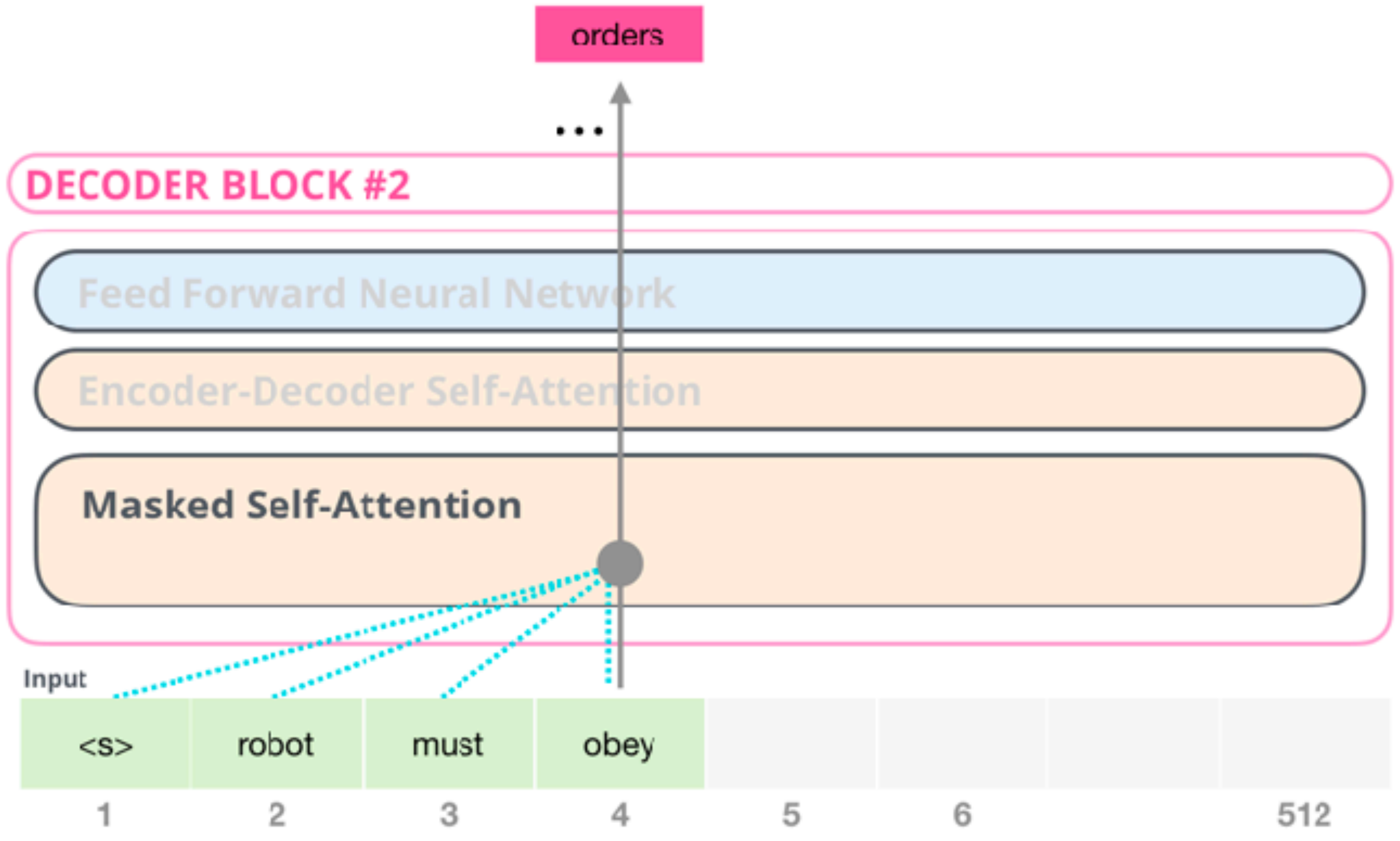
$$x_{t+1} \sim Ah_t + b$$

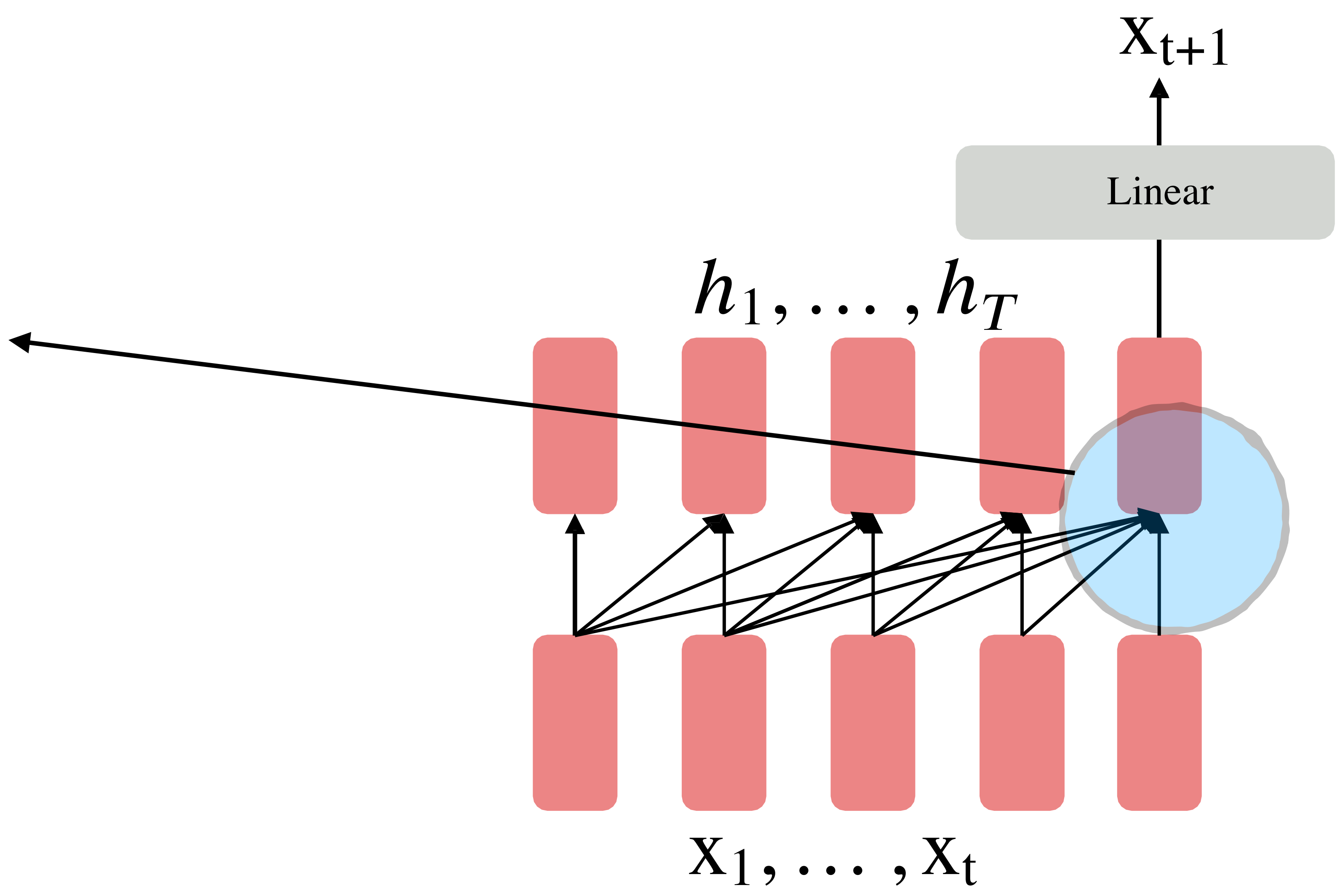
Decoder

- **GPT-models**

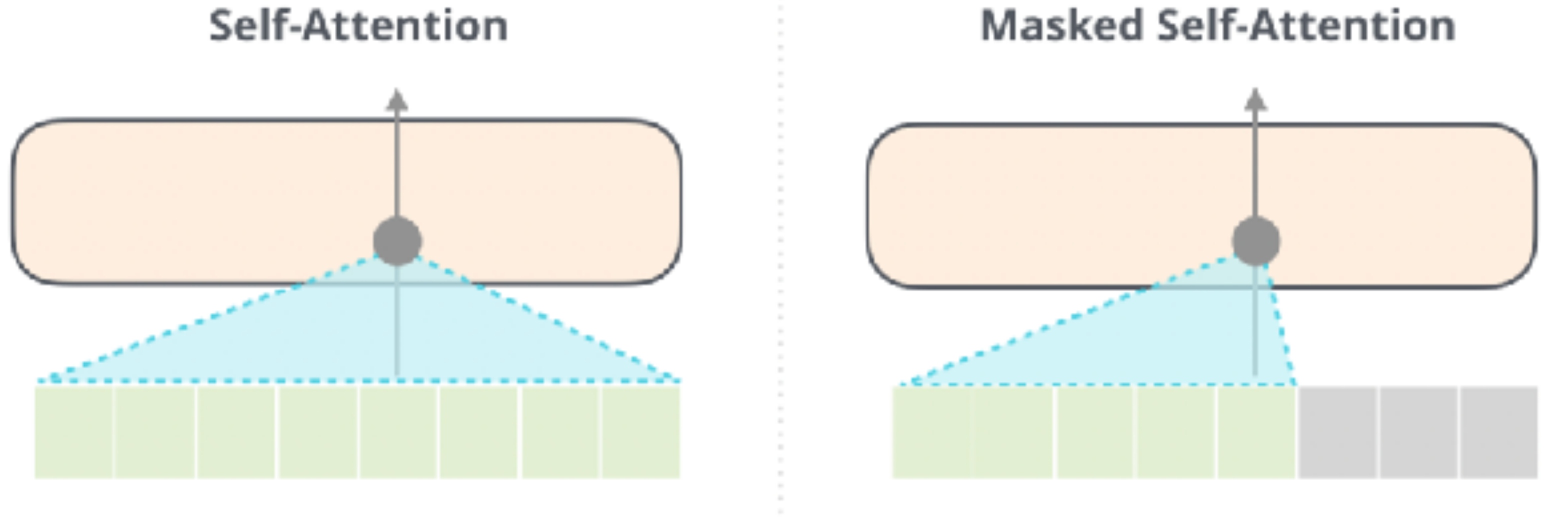
Decoder Language Model

$$\max_{\theta} \log p_\theta(x_1, \ldots, x_T) \approx \sum_{t=1}^{T} \log p_\theta(x_t \mid x_1, \ldots, x_{t-1})$$





BERT vs. GPT



$$h_1, \ldots, h_t = \text{Decoder}(x_1, \ldots, x_t)$$

$$x_{t+1} \sim A h_t + b$$

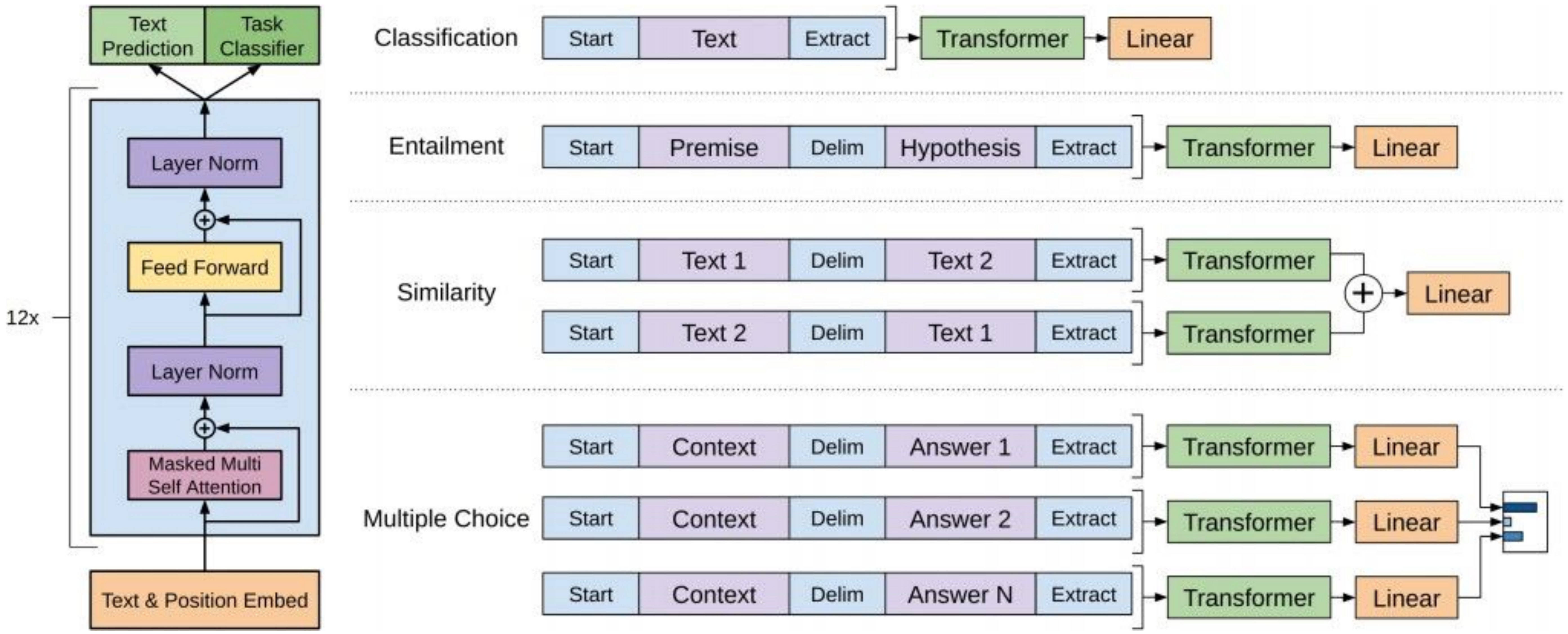
http://jalammar.github.io/illustrated-gpt2/

Decoder

- GPT-models

Generative Pre-Trained Transformer (GPT)

- Transformer decoder with 12 layers.
- Byte-pair encoding with 40,000 merges
- Trained on BooksCorpus: over 7000 unique books.
  - Contains long spans of contiguous text, for learning long-distance dependencies.

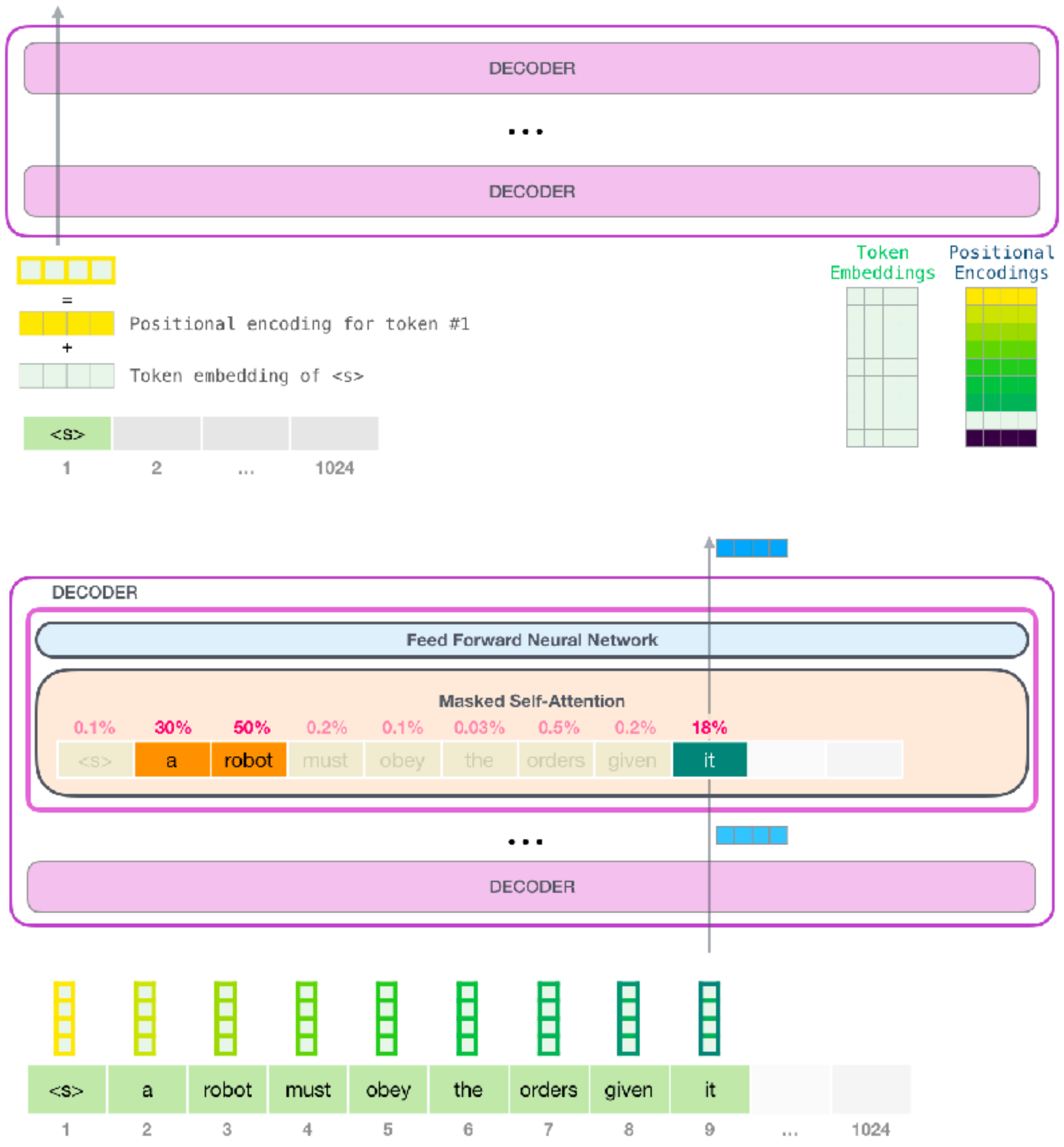


Radford et al., 2018 https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf

Decoder

- GPT-models

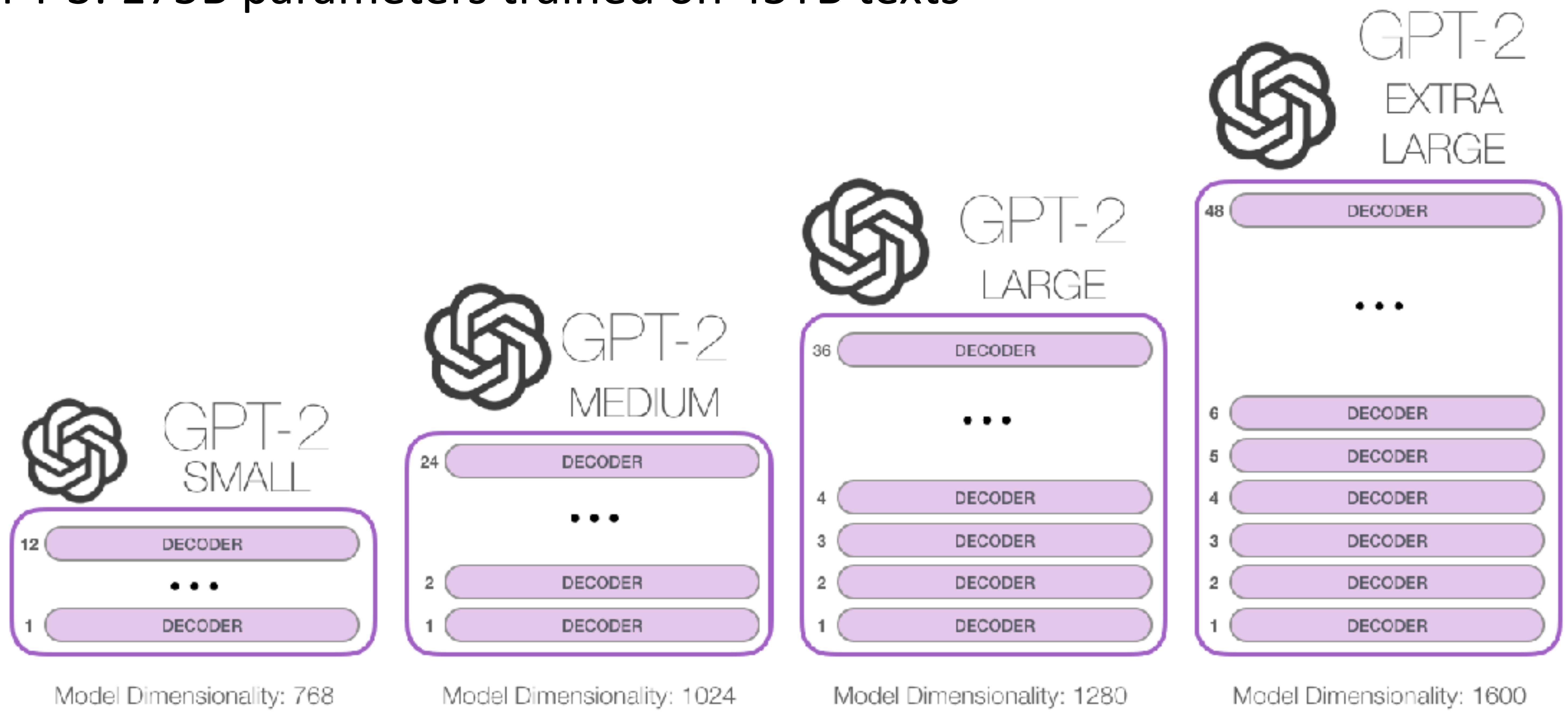
Generative Pre-Trained Transformer (GPT)



Decoder

- **GPT-models**

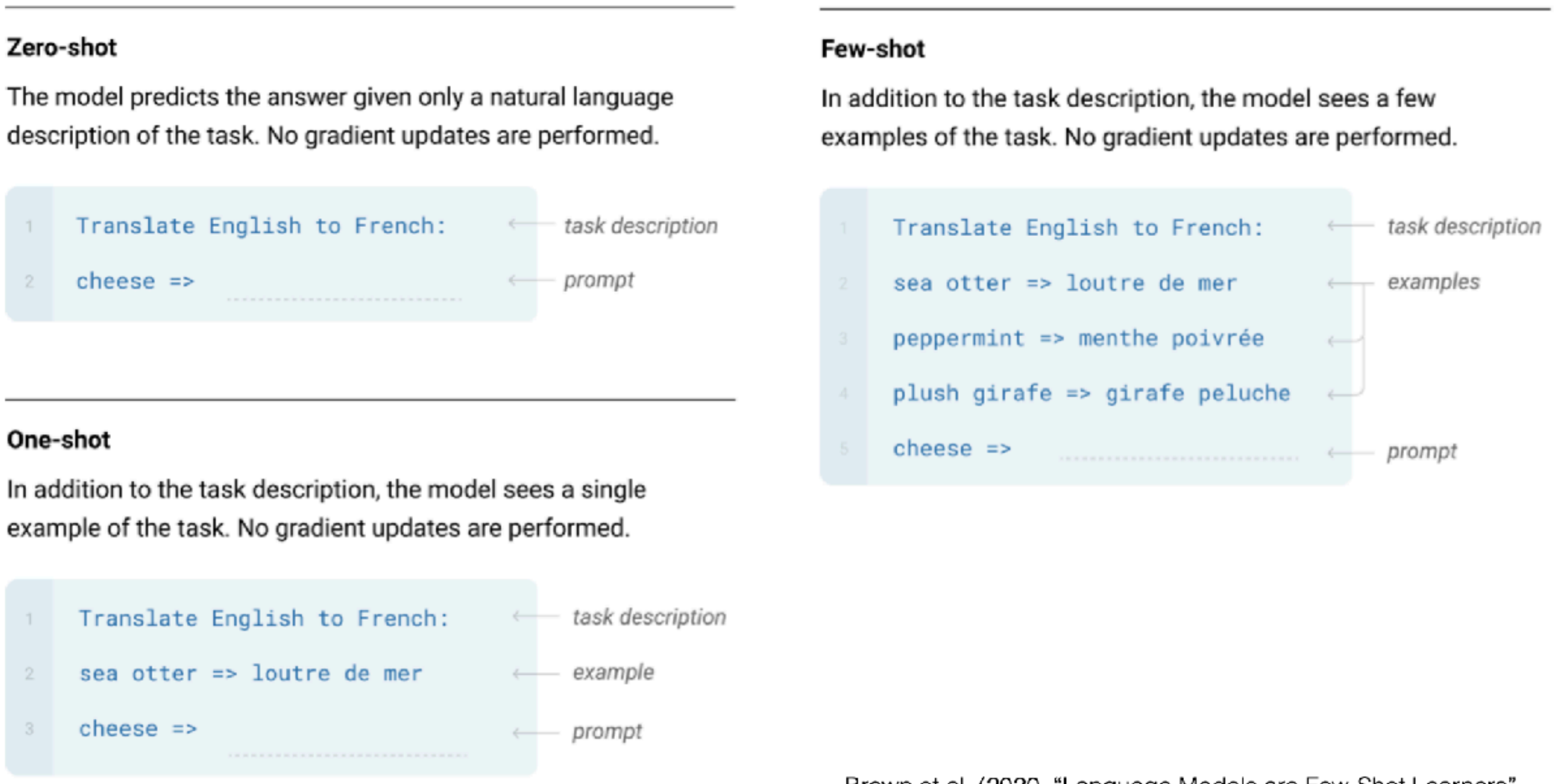
GPT released June 2018
GPT-2 released Nov. 2019 with 1.5B parameters
GPT-3: 175B parameters trained on 45TB texts



Decoder

- **GPT-models**

	Model	Data
GPT-2 (Radford et al. 2019)	Context size: 1024 tokens 117M-1.5B parameters	WebText (45 million outbound links from Reddit with 3+ karma); 8 million documents (40GB)
GPT-3 (Brown et al. 2020)	Context size: 2048 tokens 125M-175B parameters	Common crawl + WebText + “two internet-based books corpora” + Wikipedia (400B tokens, 570GB)



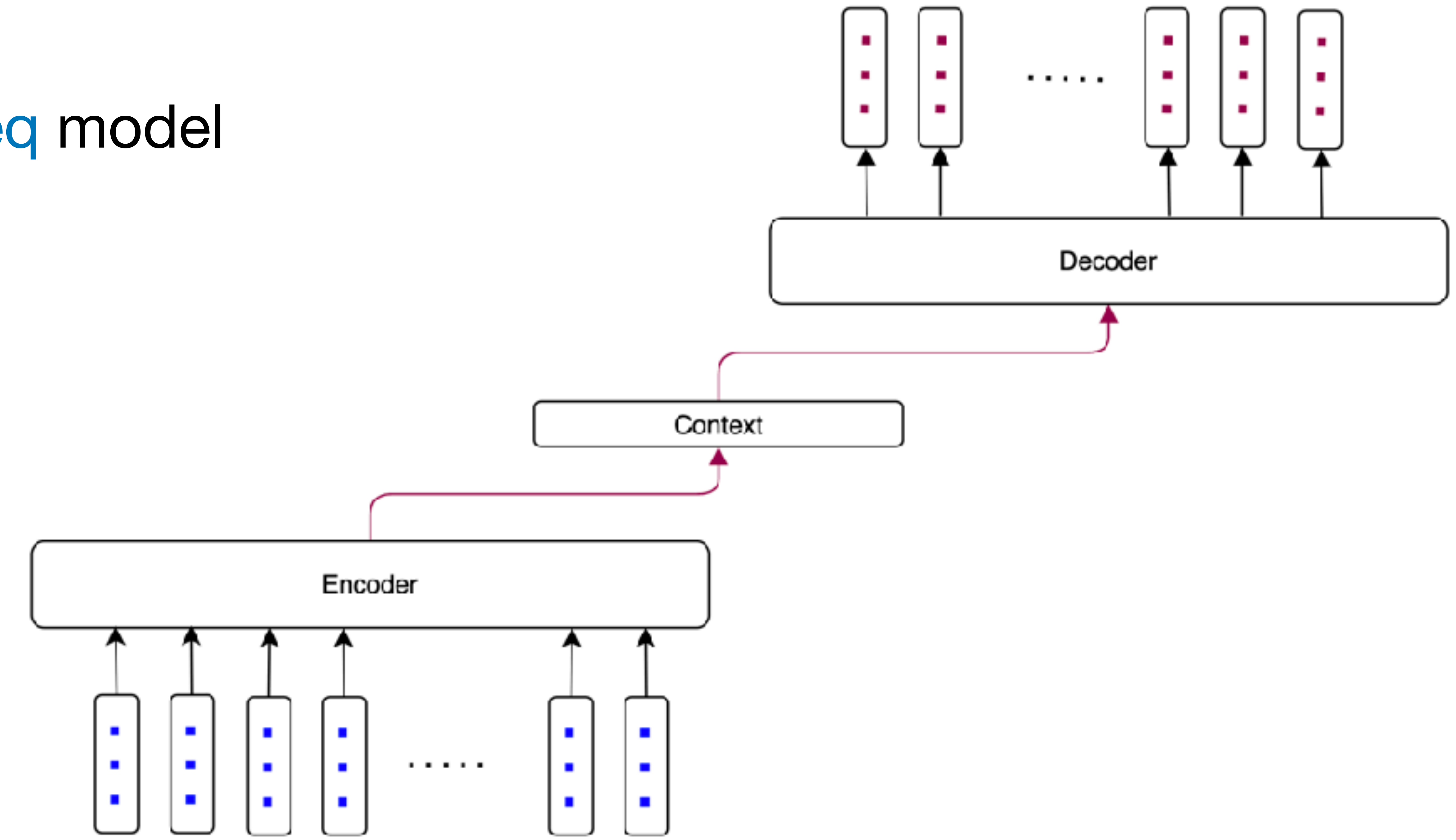
Brown et al. (2020, “Language Models are Few-Shot Learners”
https://arxiv.org/pdf/2005.14165.pdf

–Enc–Dec Language Model

- Attention is all you need,T5, BART

Basic Idea of encoder–decoder

- The encoder encodes the input into a context vector
- The decoder produces task-specific output given the context
  - Output: contextually relevant, variable-length
- Also known as seq-to-seq model

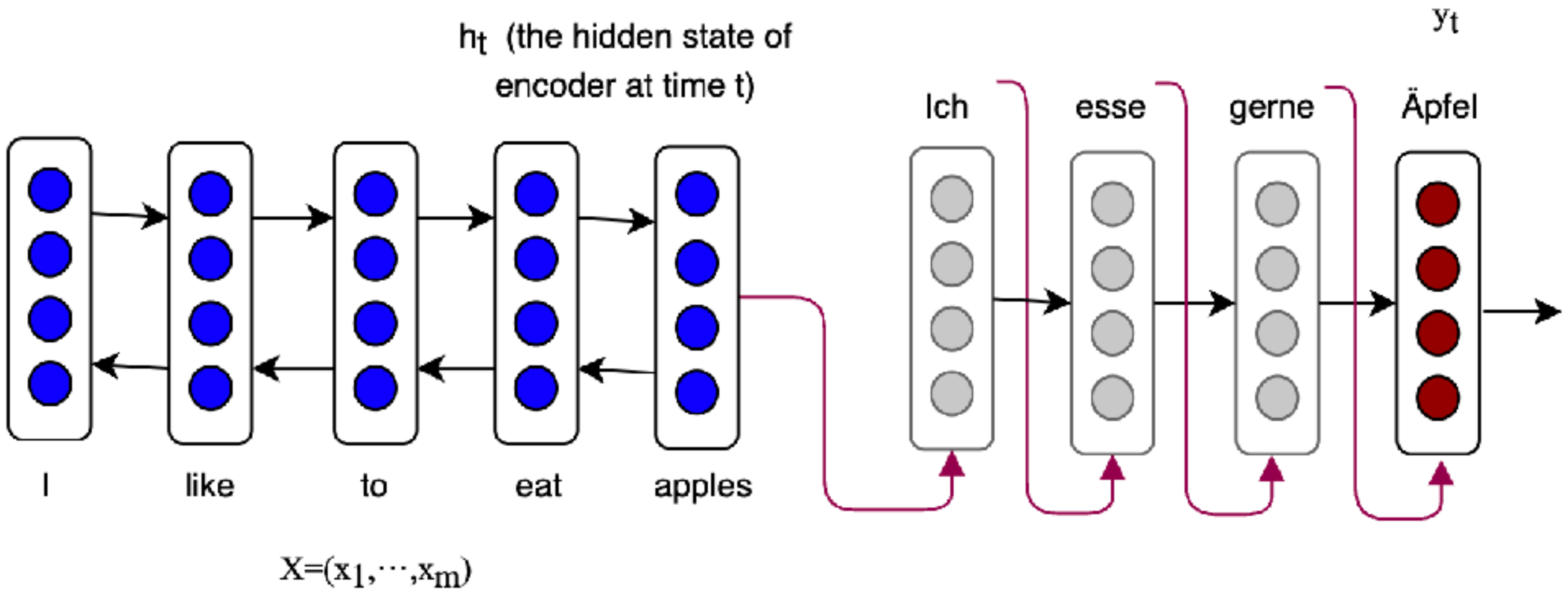


Encoder–Decoder

- **Earlier Models**
- Model T5
- Model BART
- Beam Search

Earliest works

- Machine Translation using RNNs



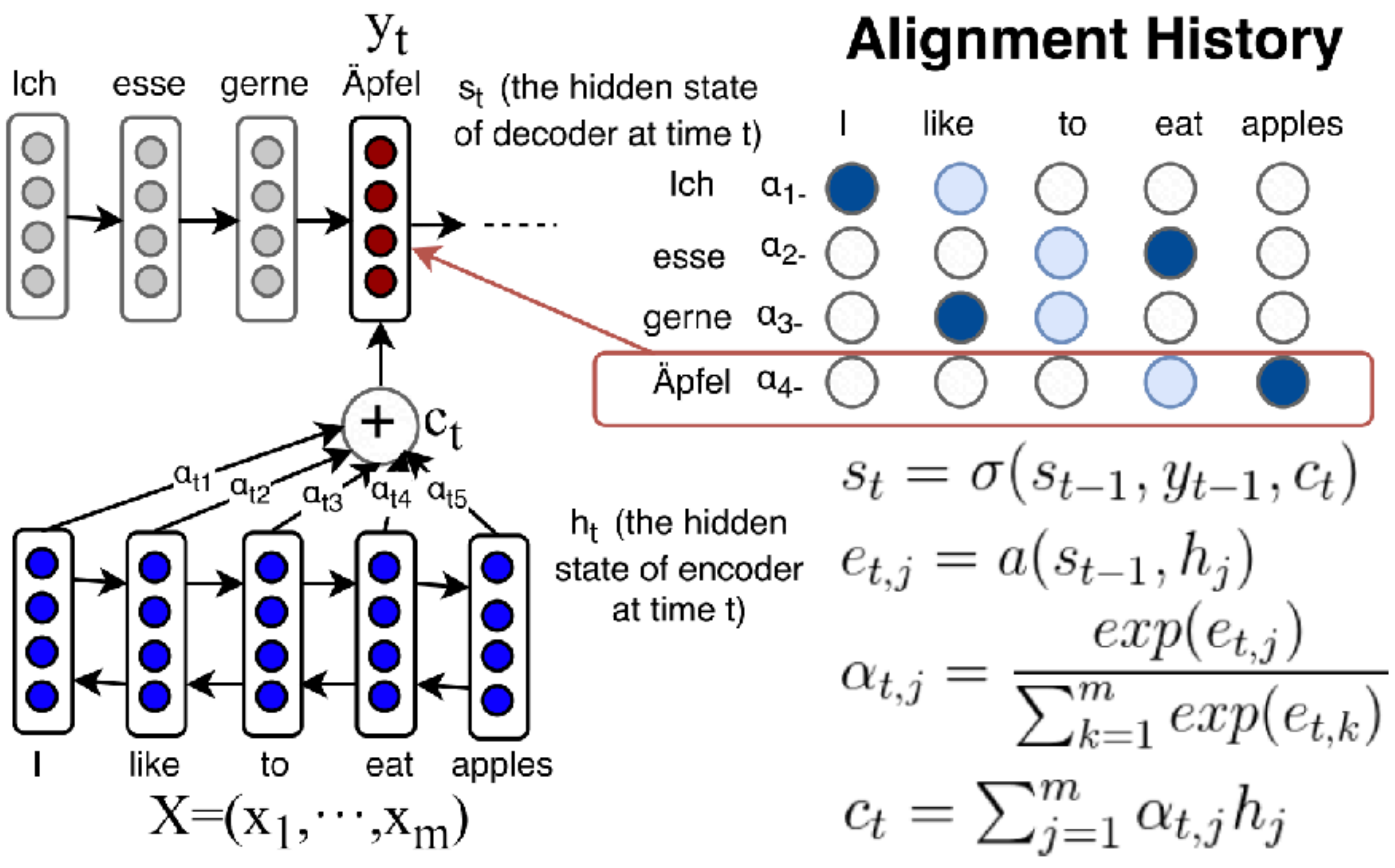
Ilya Sutskever, Oriol Vinyals, Quoc V. Le,
Sequence to Sequence Learning with Neural Networks. NIPS 2014: 3104-3112

Encoder–Decoder

- **Earlier Models**
- **Model T5**
- **Model BART**
- **Beam Search**

Earliest works

- Machine Translation using RNNs with attention mechanism



Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio,
Neural Machine Translation by Jointly Learning to Align and Translate. ICLR 2015

Encoder–Decoder

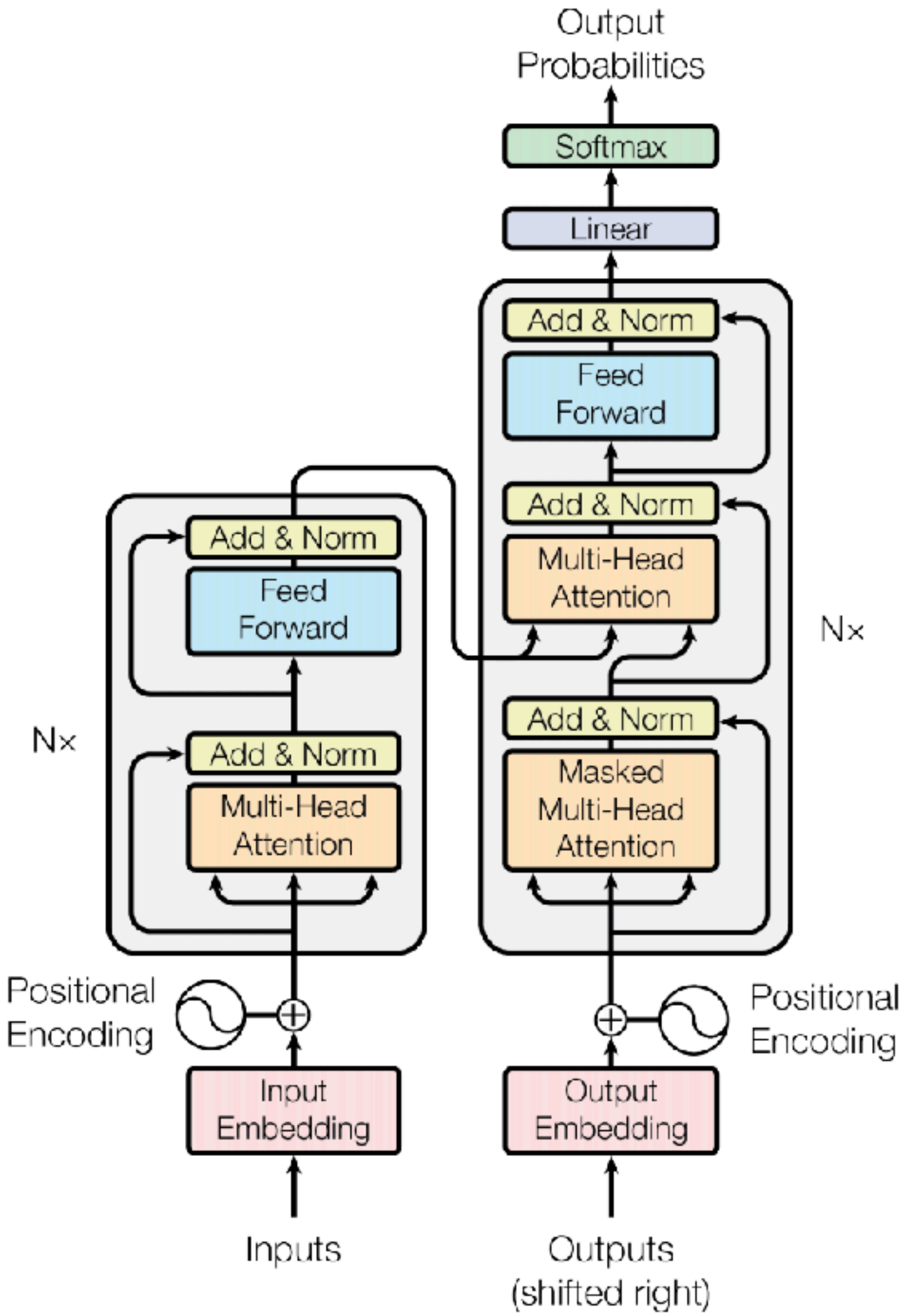
- **Earlier Models**
- **Model T5**
- **Model BART**
- **Beam Search**

Earliest works

- Introducing transformers
  - Multihead attention
  - For machine translation

$$\Pr\left(y_1, \ldots, y_n \mid \mathbf{x}\right) = \prod_i^n \Pr\left(y_i \mid y_{i-1}, \ldots, y_1, \mathbf{x}\right)$$

  - Target seq $\mathbf{y} = \left(y_1, \ldots, y_{T_y}\right)$
  - Source seq $\mathbf{x} = \left(x_1, \ldots, x_{T_x}\right)$

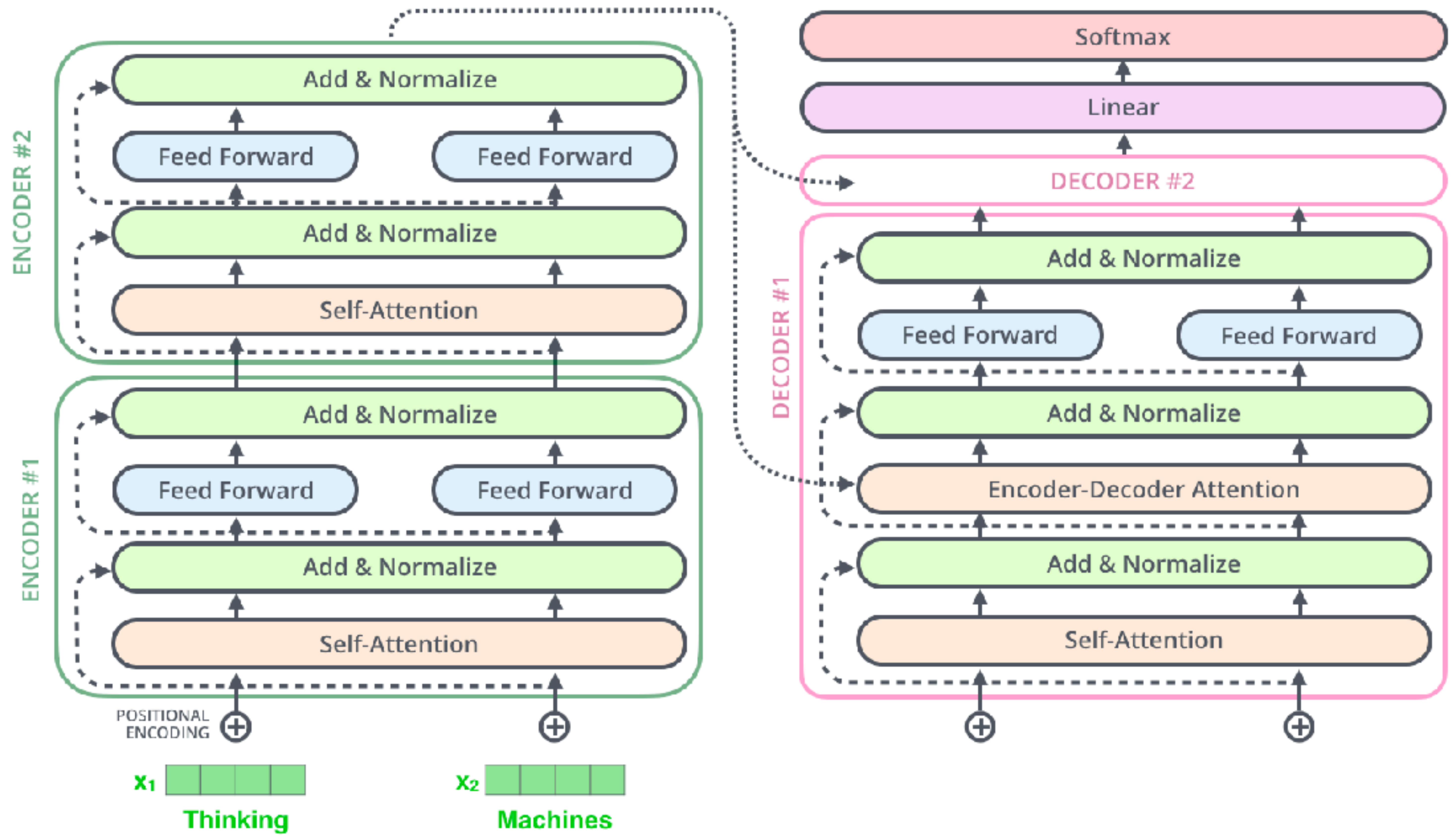


Ashish Vaswani *et al.*
Attention is All you Need. NIPS 2017: 5998-6008.

Encoder–Decoder

- Earlier Models
- Model T5
- Model BART
- Beam Search

Translation Model

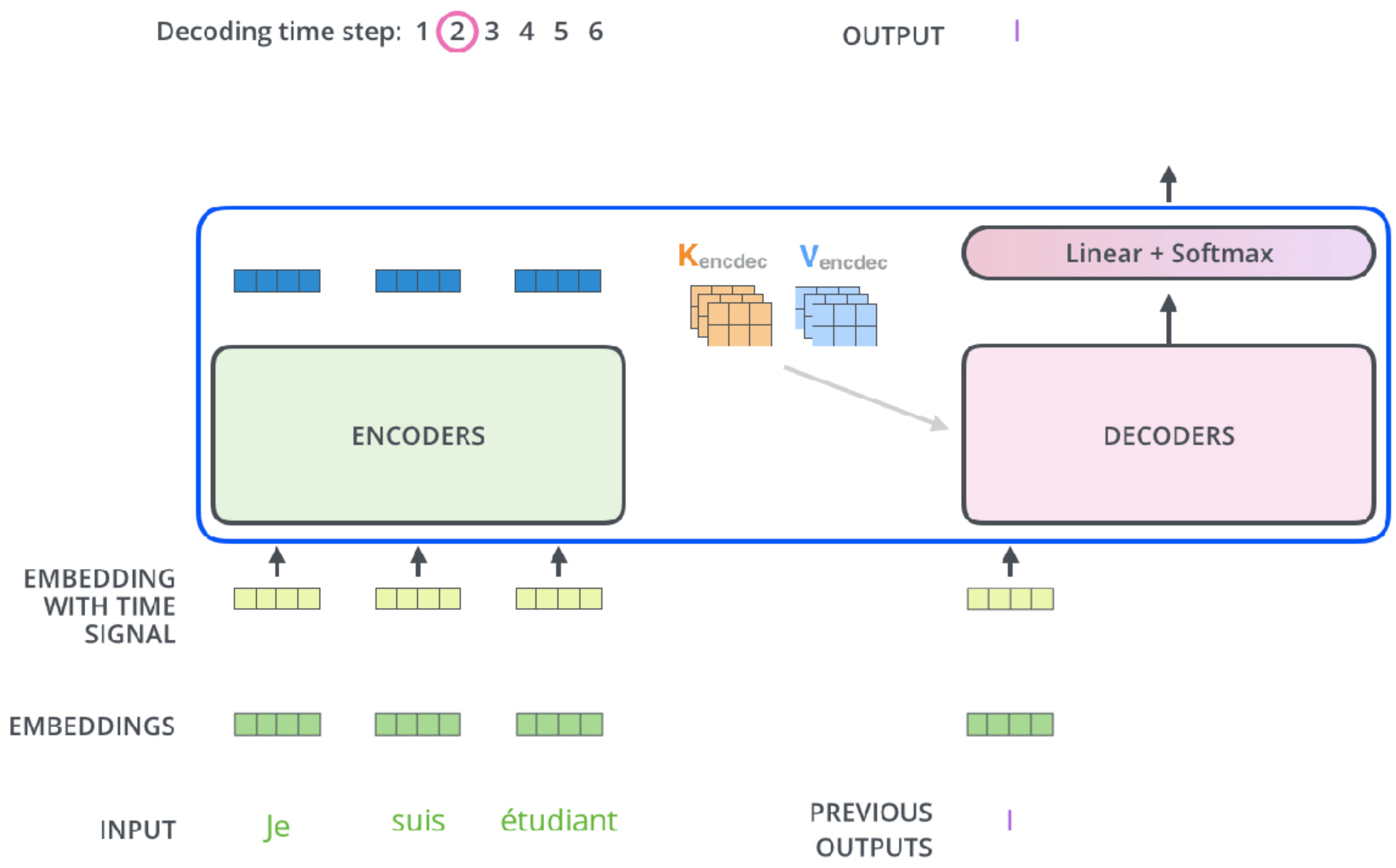


Reference: https://jalammar.github.io/

Encoder–Decoder

- **Earlier Models**
- **Model T5**
- **Model BART**
- **Beam Search**

Animation of the Translation Model



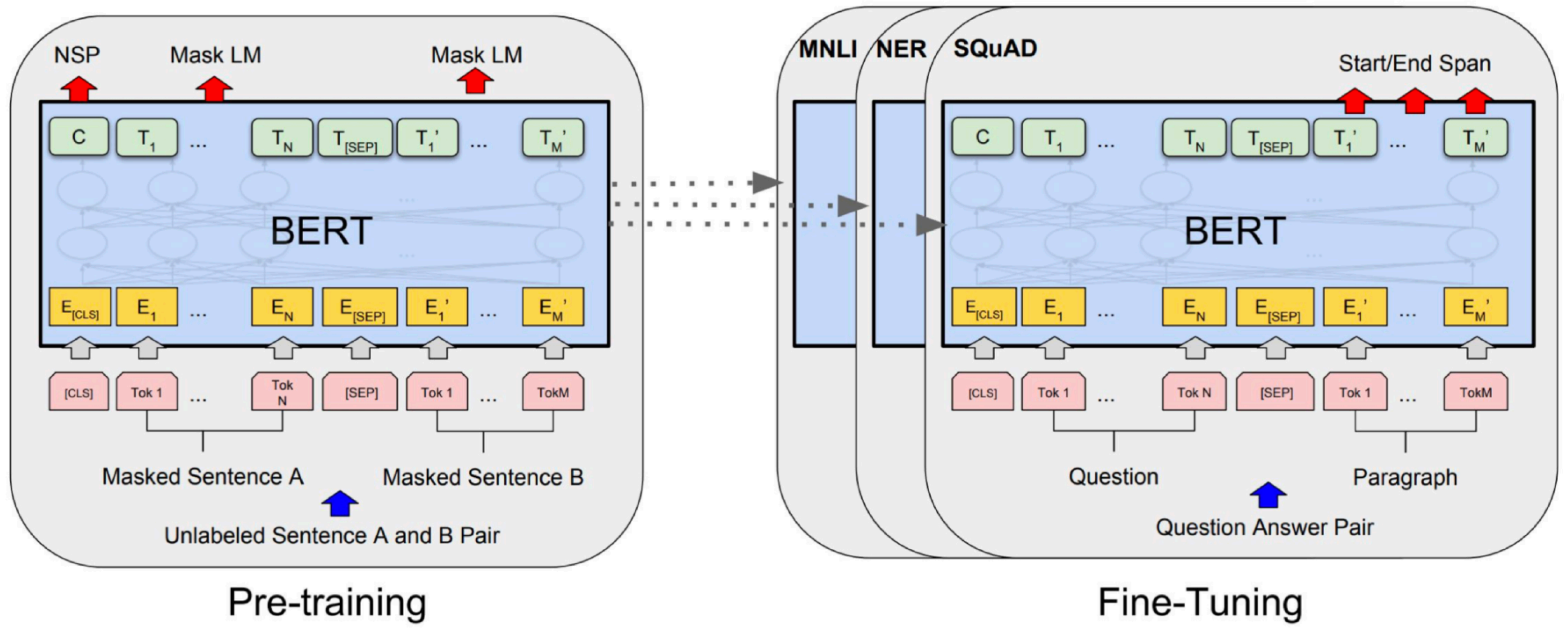
Reference: https://jalammar.github.io/

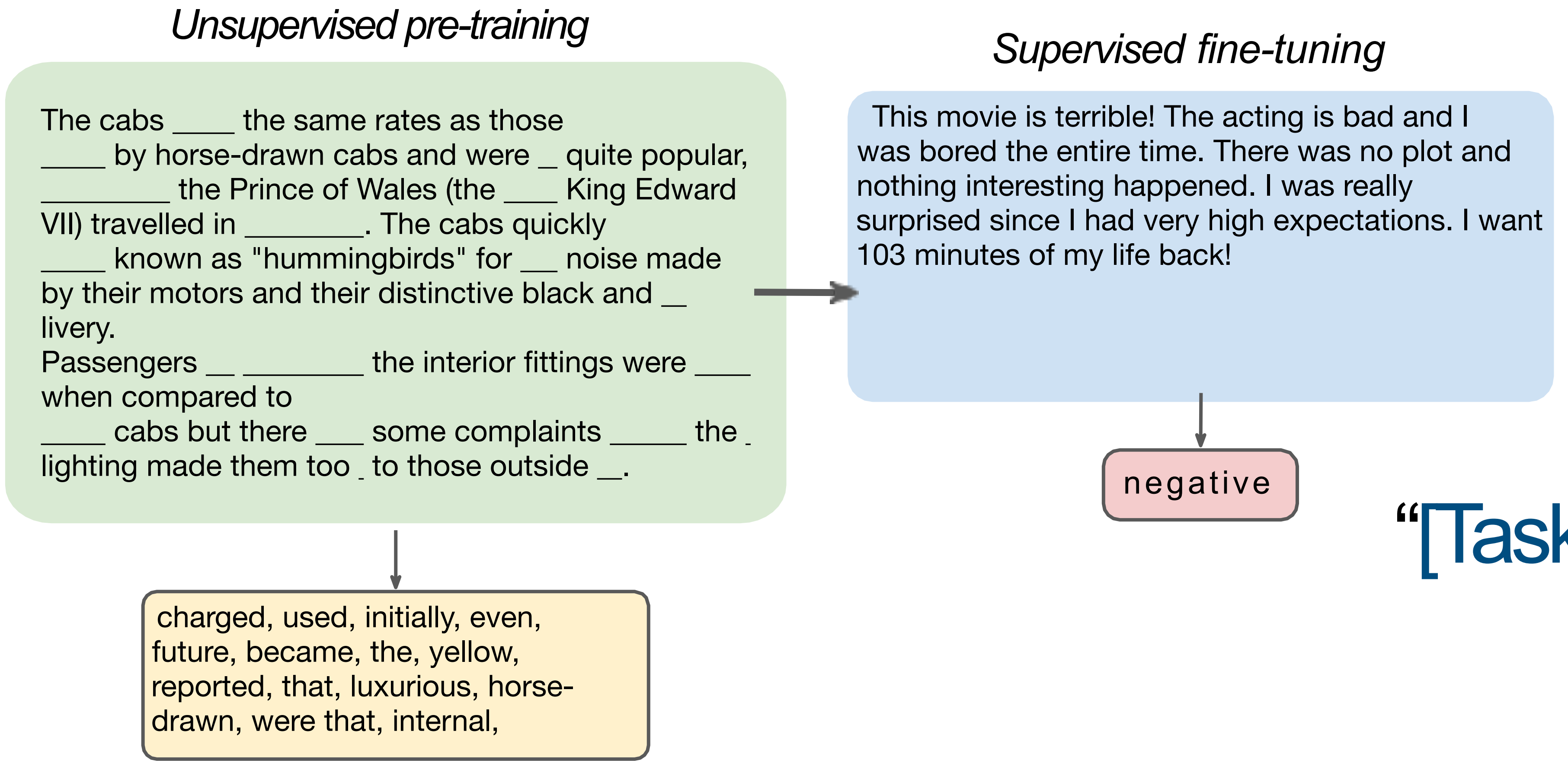
Encoder–Decoder

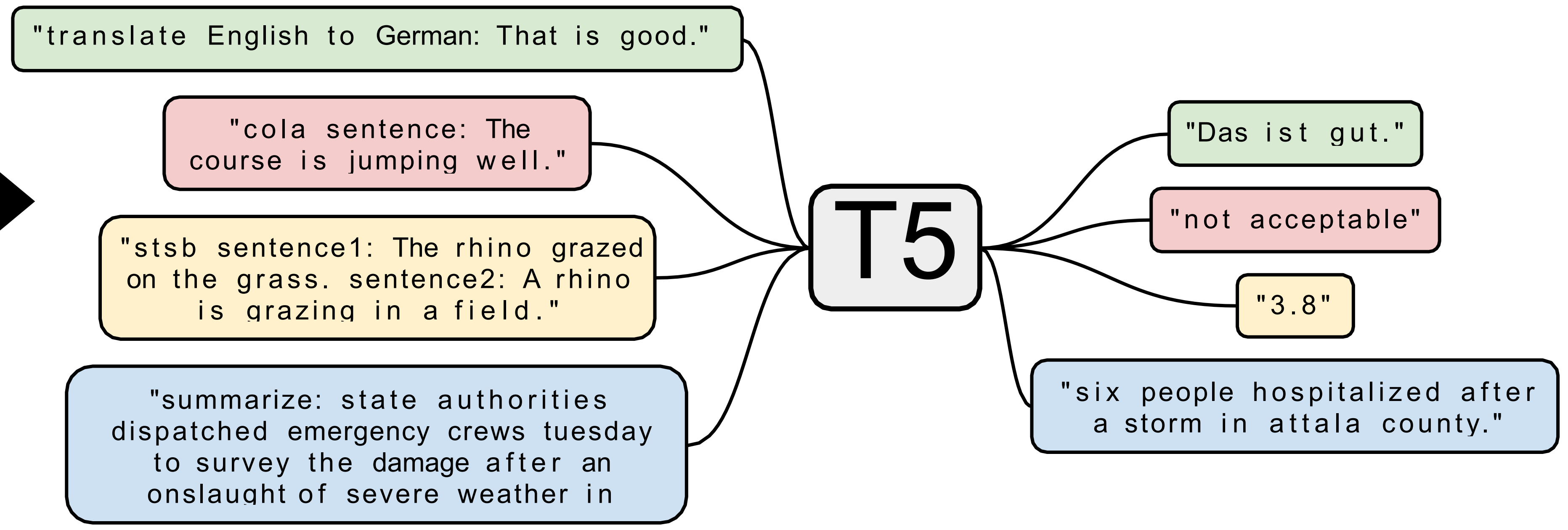
- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Basic Idea of T5

- **T**ext-to-**T**ext **T**ransfer **T**ransformer
- Moving from task-specific fine-tuning of language models ⟶ Single model for all





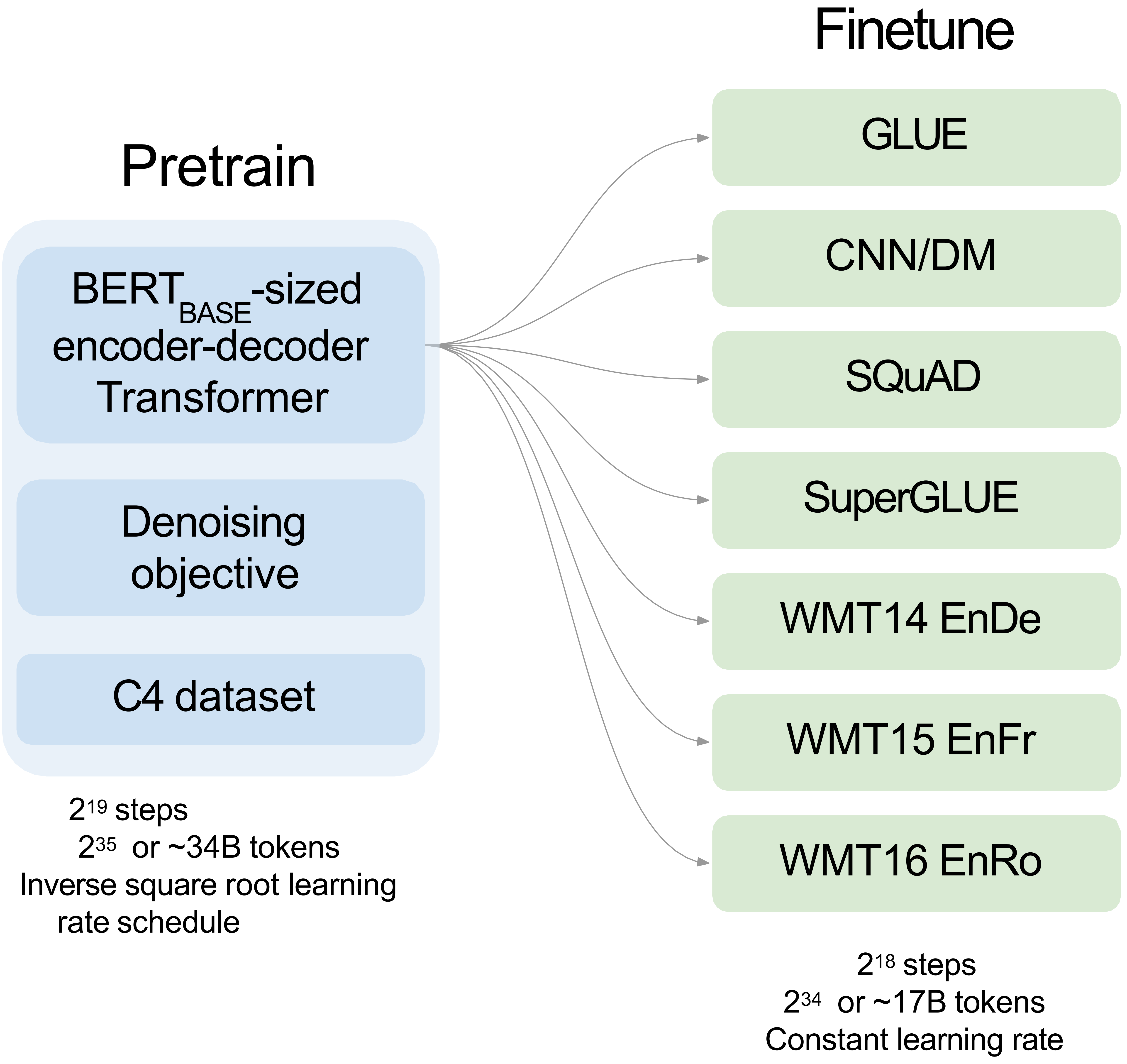


“[Task-specific prefix]: [Input text]” ⟶ “[output text]”

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Objective



Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Denoising Objective

Original text

Thank you ~~for inviting~~ me to your party ~~last~~ week.

Inputs

Thank you <X> me to your party <Y> week.

Targets

<X> for inviting <Y> last <Z>

Encoder–Decoder

- Earlier Models
- Model T5
- Model BART
- Beam Search

Finetuning Examples

- CoLA (GLUE): Sentence acceptability
  - **Input**: sentence, **output**: labels "acceptable" or "not acceptable"
  - Ex: "The course is jumping well." -> not acceptable
- STS-B (GLUE): Sentence similarity
  - **Input**: pair of sentences, **output**: similarity score [1,5]
  - Ex: "sentence1: The rhino grazed. sentence2: A rhino is grazing." -> 3.8
- COPA (SuperGLUE): Causal reasoning
  - **Input**: premise and 2 alternatives, **output**: alternative1 or alternative2
  - Ex: "Premise: I tipped the bottle. What happened as a RESULT? Alternative 1: The liquid in the bottle froze. Alternative 2: The liquid in the bottle poured out." -> alternative2
- EnDe (Translation):

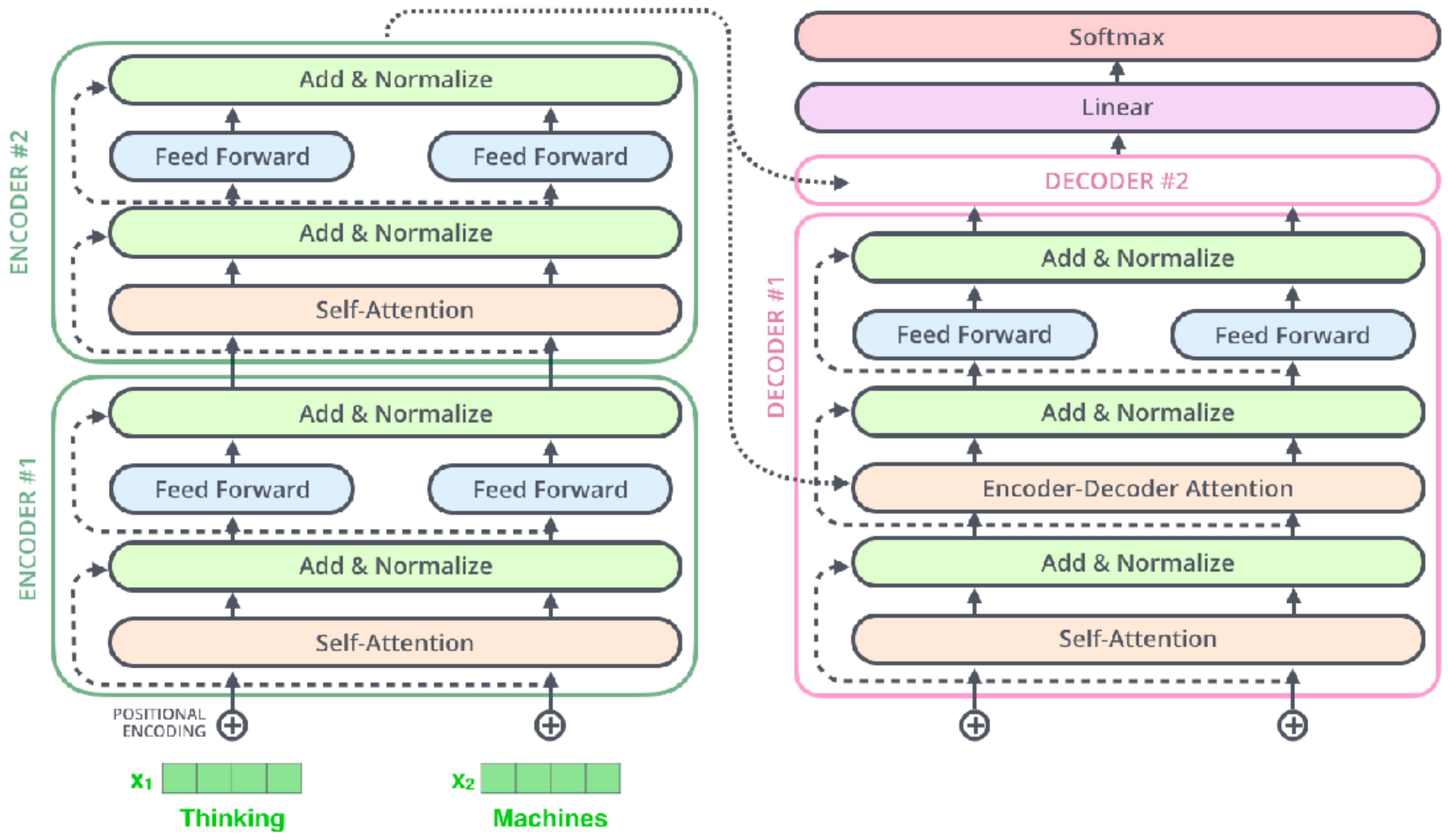
  "translate English to German: That is good" -> "Das ist gut"
- CNNDM (Summarization):
- "summarize: state authorities dispatched…" -> "six people hospitalized after storm"

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Vocab

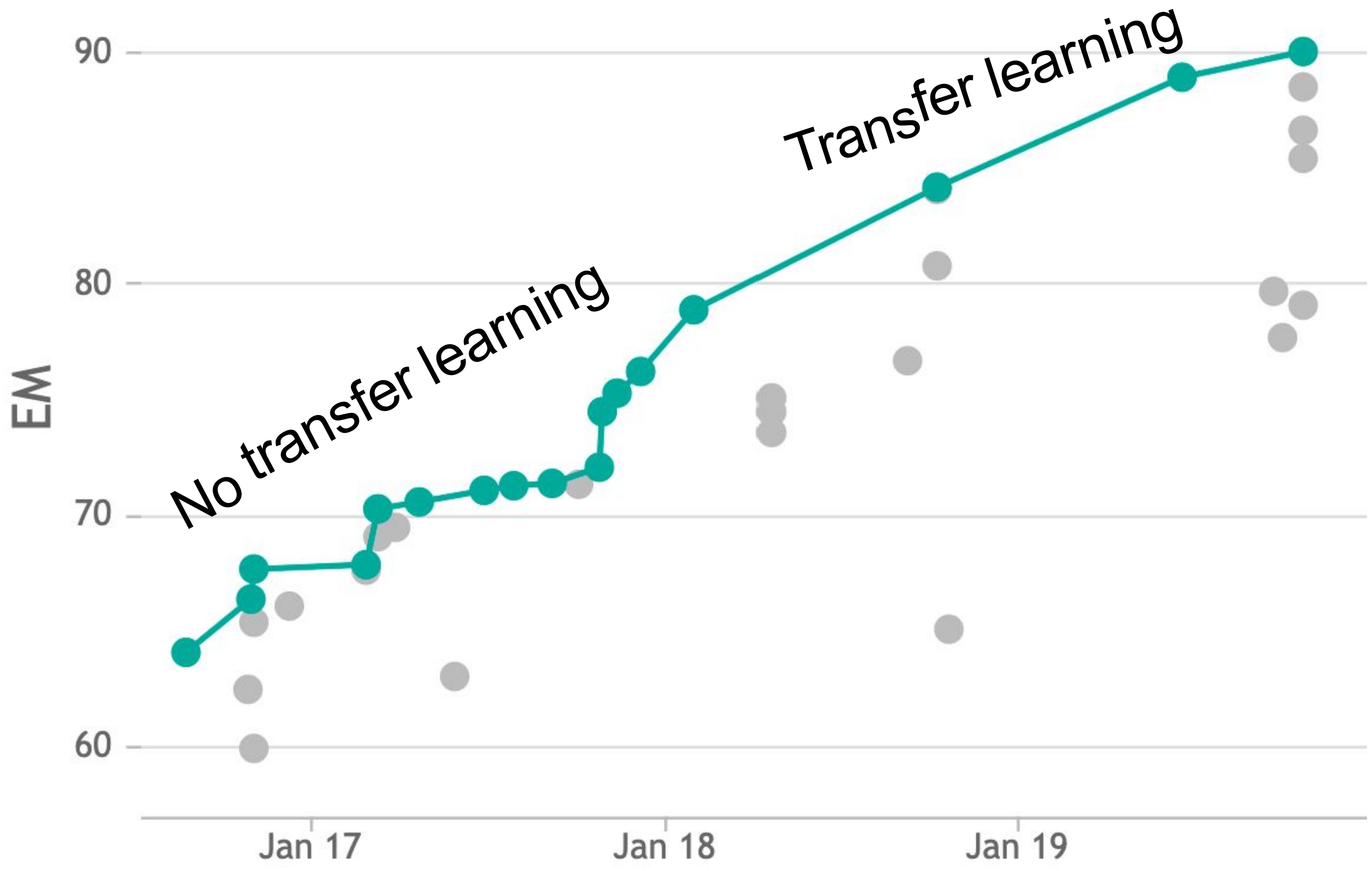
- 32,000 wordpieces shared across input and output
- Pre-training is English, but fine-tuning includes German, French, and Romanian



Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Every task is Question-Answering

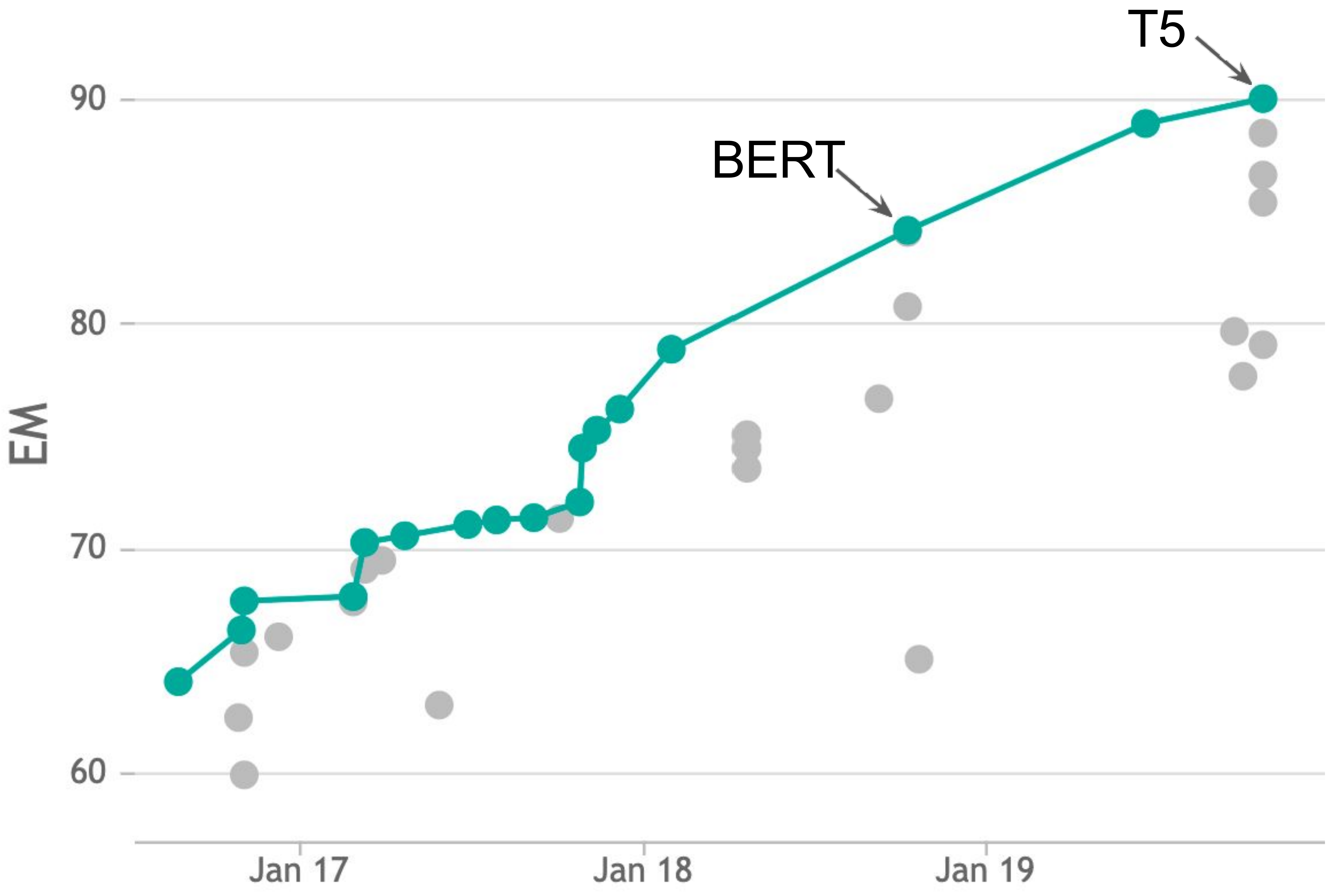


Source: https://paperswithcode.com/sota/question-answering-on-squad11-dev

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Every task is Question-Answering



Source: https://paperswithcode.com/sota/question-answering-on-squad11-dev

Training Dataset

- **C4 Dataset: C**olossal **C**lean **C**rawled **C**orpus
- Web-extracted text
- English language only (langdetect)
- Extreme cleaning and filtering: 20TB → 750GB

Menu

The lemon, Citrus Limon (l.) Osbeck, is a species of small evergreen tree in the flowering plant family rutaceae.
The tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses.
The juice of the lemon is about 5% to 6% citric acid, with a ph of around 2.2, giving it a sour taste.

Article

The origin of the lemon is unknown, though lemons are thought to have first grown in Assam (a region in northeast India), northern Burma or China.
A genomic study of the lemon indicated it was a hybrid between bitter orange (sour orange) and citron.

Please enable JavaScript to use our site.

Dried Lemons, $3.59/pound

Organic dried lemons from our farm in California.
Lemons are harvested and sun-dried for maximum flavor.
Good in soups and on popcorn.

The lemon, Citrus Limon (l.) Osbeck, is a species of small evergreen tree in the flowering plant family rutaceae.
The tree's ellipsoidal yellow fruit is used for culinary and non-culinary purposes throughout the world, primarily for its juice, which has both culinary and cleaning uses.
The juice of the lemon is about 5% to 6% citric acid, with a ph of around 2.2, giving it a sour taste.

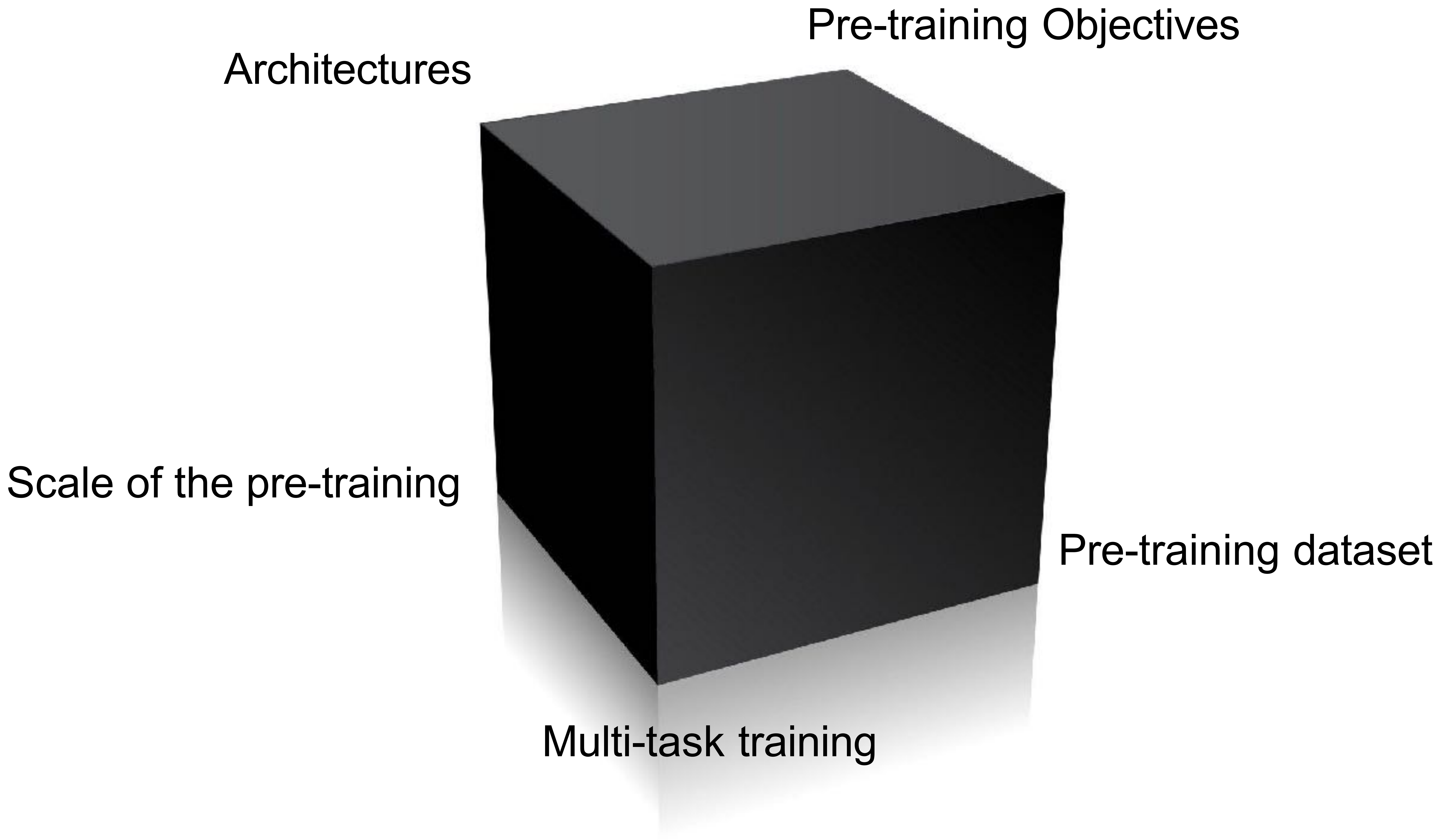
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Curabitur in tempus quam. In mollis et ante at consectetur.
Aliquam erat
volutpat.
Donec at
lacinia est.
Duis semper, magna tempor interdum suscipit, ante elit molestie urna, eget efficitur risus nunc ac elit.
Fusce quis blandit lectus.
Mauris at mauris a turpis tristique lacinia at nec ante.
Aenean in scelerisque tellus, a efficitur ipsum.
Integer justo enim, ornare vitae sem non, mollis fermentum lectus.
Mauris ultrices nisl at libero porta sodales in ac orci.

```
function Ball(r)
    {this.radius = r;
```

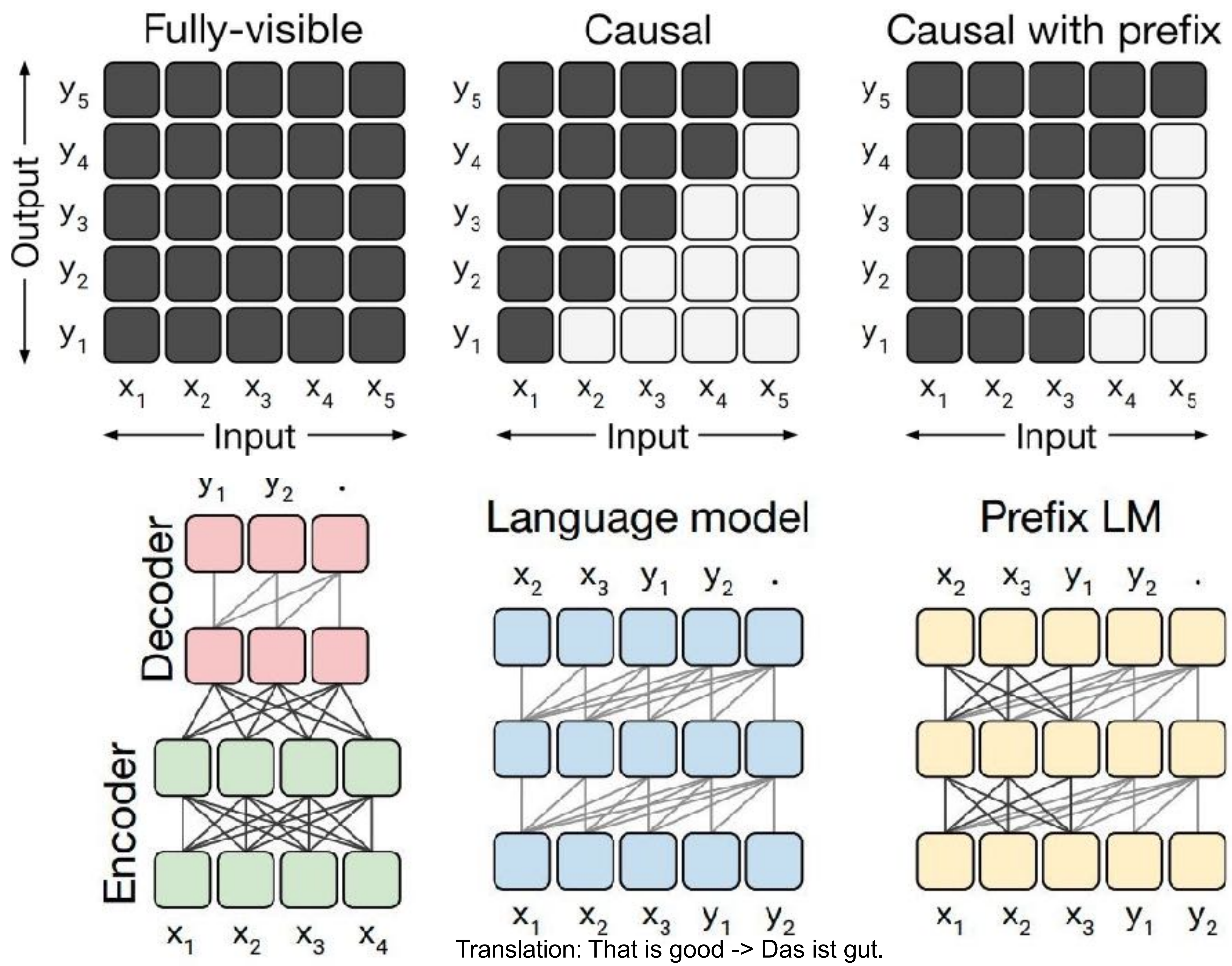
Trying different decisions for Pre-training and Fine-tuning

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search



Architecture - Attention Mask



Translation: That is good -> Das ist gut.

Translate English to German: That is good. Target: Das is gut.

Translate English to German: That is good. Target: Das is gut.
"Good" representation can look at "Translate English to German: That is. Target:".

Translate English to German: That is good. Target: Das is gut.
"Good" representation can only look at "Translate English to German: That is".

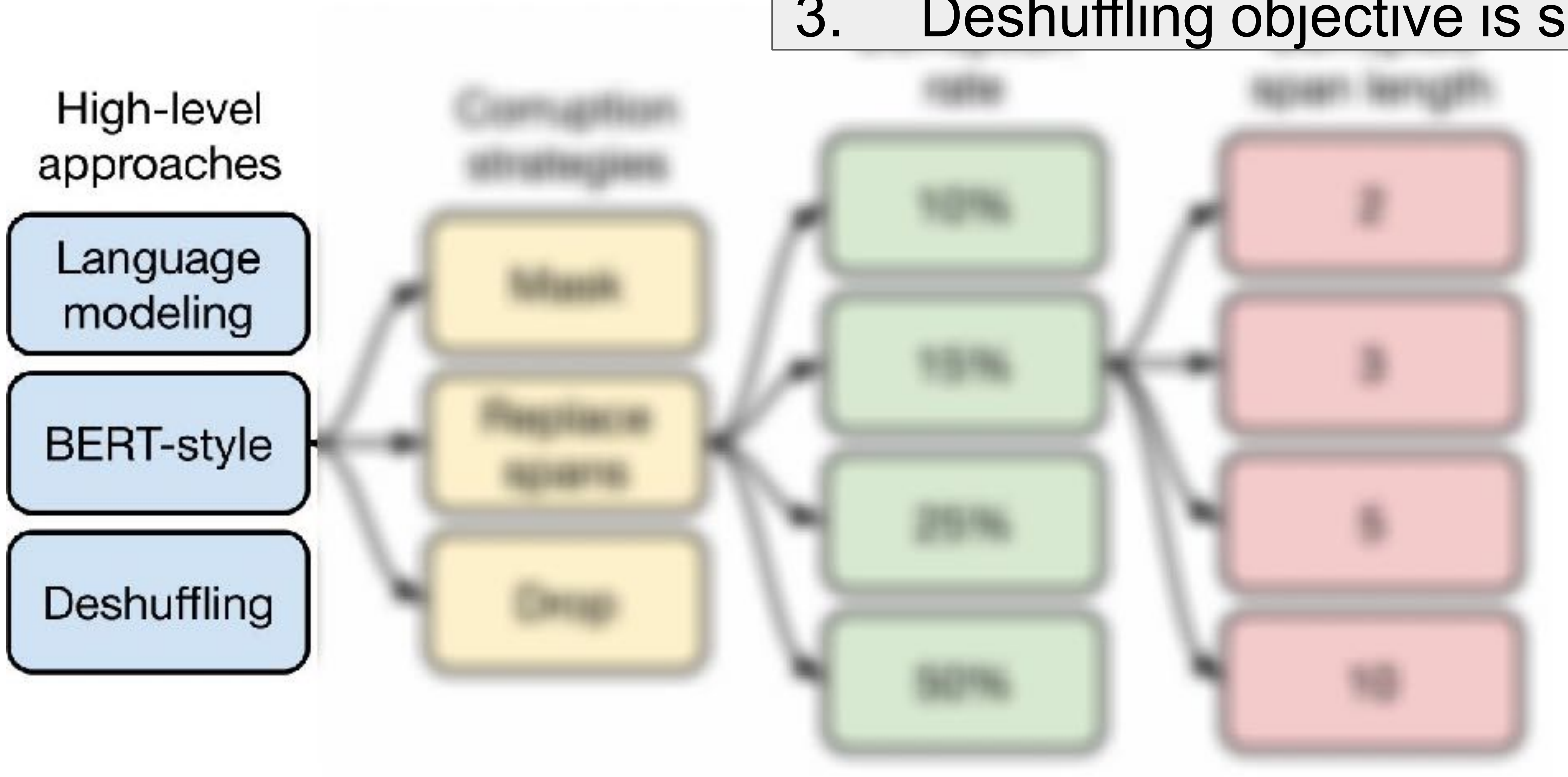
Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Pretraining Objective

1. BERT-style objective performs best.
2. Prefix LM works well on translation tasks.
3. Deshuffling objective is significantly worse.



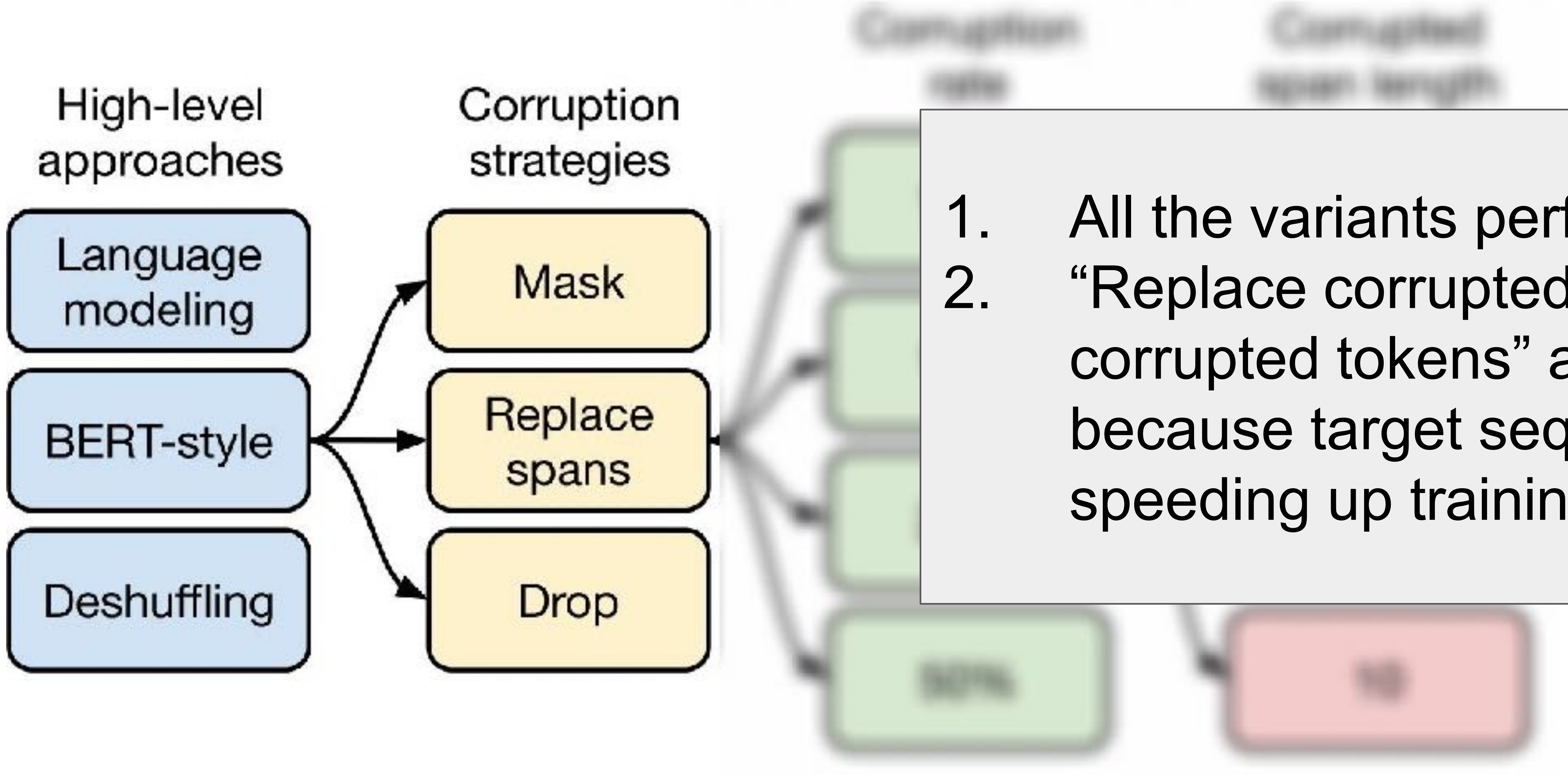
Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERT-style (Devlin et al., 2018)	**82.96**	**19.17**	**80.65**	**69.85**	**26.78**	**40.03**	**27.41**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62

Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*

Pretraining Objective

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search



1. All the variants perform similarly.
2. “Replace corrupted spans” and “Drop corrupted tokens” are more appealing because target sequences are shorter, speeding up training.

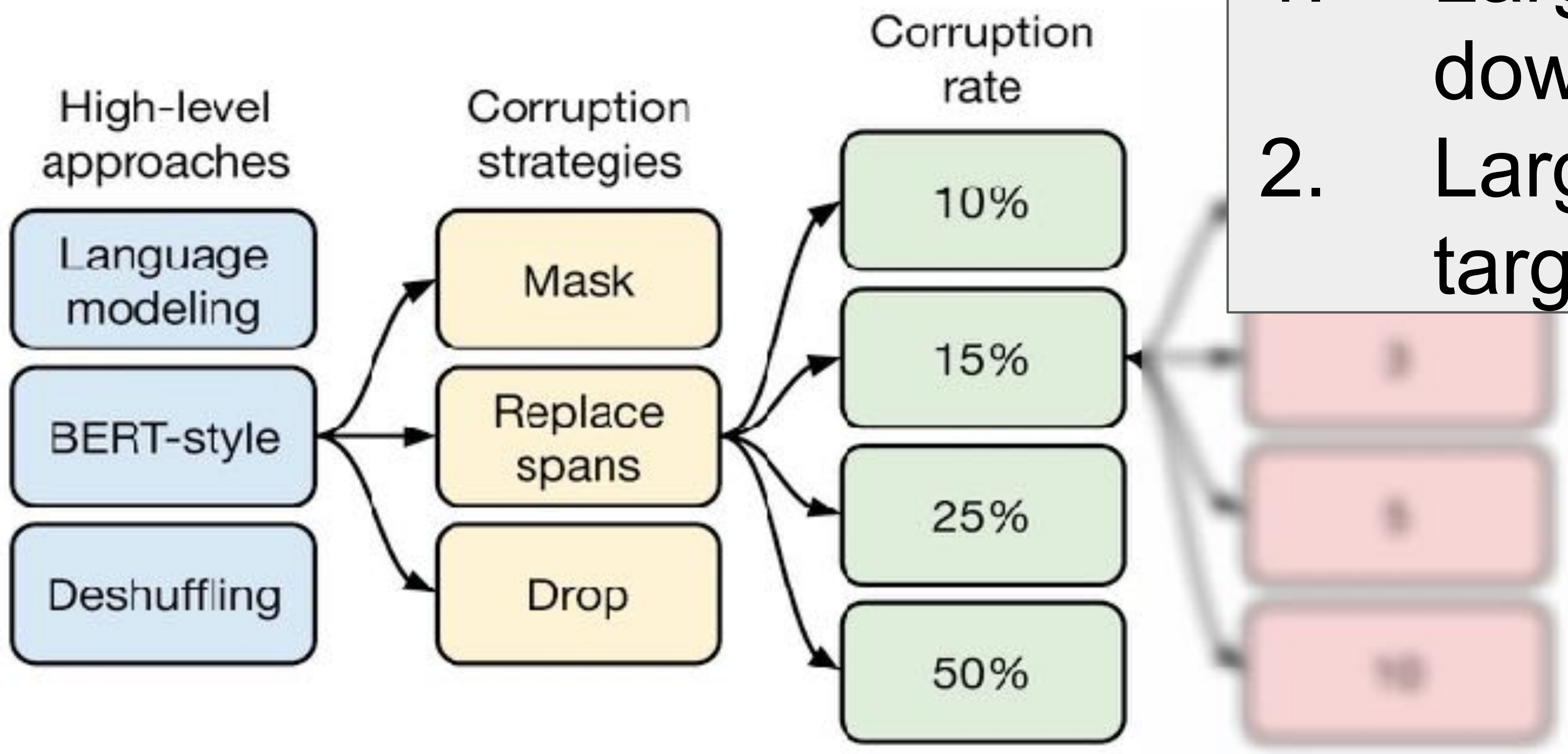
Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
BERT-style (Devlin et al., 2018)	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASS-style (Song et al., 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
★ Replace corrupted spans	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
Drop corrupted tokens	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Different Corruption Rates

1. Larger corruption rate leads to downstream performance degradation.
2. Larger corruption rate also leads to longer targets, slowing down training.

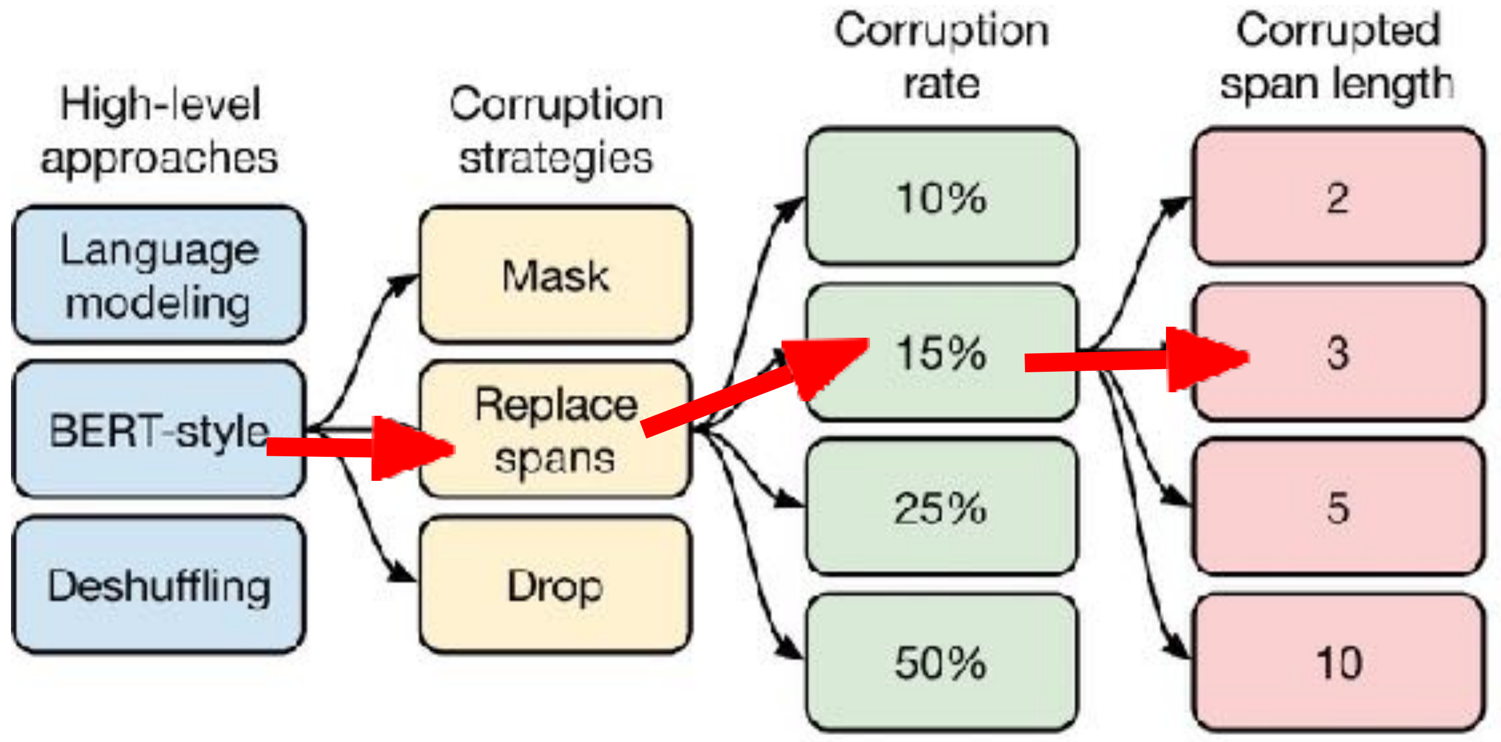


Corruption rate	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
10%	**82.82**	19.00	**80.38**	69.55	**26.87**	39.28	**27.44**
★ 15%	**83.28**	19.24	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
25%	**83.00**	**19.54**	**80.96**	70.48	**27.04**	**39.83**	**27.47**
50%	81.27	19.32	79.80	70.33	**27.01**	**39.90**	**27.49**

Span-corruption rate

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search



1. Average span length of 3 works well on most non-translation tasks.
2. Span corruption produces shorter target sequences and leads to speedup in training.

Span length	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (i.i.d.)	**83.28**	19.24	80.88	71.36	**26.98**	**39.82**	**27.65**
2	**83.54**	19.39	**82.09**	**72.20**	**26.76**	**39.99**	**27.63**
3	**83.49**	**19.62**	**81.84**	**72.53**	**26.86**	39.65	**27.62**
5	**83.40**	19.24	**82.05**	**72.23**	**26.88**	39.40	**27.53**
10	82.85	19.33	**81.84**	70.44	**26.79**	39.49	**27.69**

Multitasking

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
Supervised multi-task pre-training	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**

1. Multi-task pre-training + fine-tuning works as well as unsupervised pre-training + fine-tuning.
2. Practical benefit of Multi-task pre-training + fine-tuning is to monitor downstream performance during pre-training.

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Architecture	Objective	Params	Cost	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Encoder-decoder	Denoising	$2P$	M	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec, shared	Denoising	P	M	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec, 6 layers	Denoising	P	$M/2$	80.88	18.97	77.59	68.42	26.38	38.40	26.95
Language model	Denoising	P	M	74.70	17.93	61.14	55.02	25.09	35.28	25.86
Prefix LM	Denoising	P	M	81.82	18.61	78.94	68.11	26.43	37.98	27.39

Span length	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline (i.i.d.)	**83.28**	19.24	80.88	71.36	**26.98**	**39.82**	**27.65**
2	**83.54**	19.39	**82.09**	**72.20**	**26.76**	**39.99**	**27.63**
3	**83.49**	**19.62**	**81.84**	**72.53**	**26.86**	39.65	**27.62**
5	**83.40**	19.24	**82.05**	**72.23**	**26.88**	39.40	**27.53**
10	82.85	19.33	**81.84**	70.44	**26.79**	39.49	**27.69**

Data set	Size	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
C4, unfiltered	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
RealNews-like	35GB	**83.83**	**19.23**	80.39	72.38	**26.75**	**39.90**	**27.48**
WebText-like	17GB	**84.03**	**19.31**	**81.42**	71.40	**26.80**	**39.74**	**27.59**
Wikipedia	16GB	81.85	**19.31**	81.29	68.01	**26.94**	39.69	**27.67**
Wikipedia + TBC	20GB	83.65	**19.28**	**82.08**	**73.24**	**26.77**	39.63	**27.57**

Training strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Unsupervised pre-training + fine-tuning	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
Multi-task training	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
Multi-task pre-training + fine-tuning	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
Leave-one-out multi-task training	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
Supervised multi-task pre-training	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**

Scaling strategy	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
★ Baseline	83.28	19.24	80.88	71.36	26.98	39.82	27.65
1× size, 4× training steps	85.33	19.33	82.45	74.72	27.08	40.66	27.93
1× size, 4× batch size	84.60	19.42	82.52	74.64	27.07	40.60	27.84
2× size, 2× training steps	**86.18**	19.66	**84.18**	77.18	27.52	**41.03**	28.19
4× size, 1× training steps	**85.91**	19.73	**83.86**	**78.04**	27.47	40.71	28.10
4× ensembled	84.77	**20.10**	83.09	71.74	**28.05**	40.53	**28.57**
4× ensembled, fine-tune only	84.05	19.57	82.36	71.55	27.55	40.22	28.09

Model Size

Encoder–Decoder

- Earlier Models
- **Model T5**
- Model BART
- Beam Search

Model	Parameters	No. of layers	d_{model}	d_{ff}	d_{kv}	No. of heads
Small	60M	6	512	2048	64	8
Base	220M	12	768	3072	64	12
Large	770M	24	1024	4096	64	16
3B	3B	24	1024	16384	128	32
11B	11B	24	1024	65536	128	128

Model	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Previous best	89.4	20.30	95.5	84.6	**33.8**	**43.8**	**38.5**
T5-Small	77.4	19.56	87.24	63.3	26.7	36.0	26.8
T5-Base	82.7	20.34	92.08	76.2	30.9	41.2	28.0
T5-Large	86.4	20.68	93.79	82.3	32.0	41.5	28.1
T5-3B	88.5	21.02	94.95	86.4	31.8	42.6	28.2
T5-11B	**89.7**	**21.55**	**95.64**	**88.9**	32.1	43.4	28.1

Encoder–Decoder

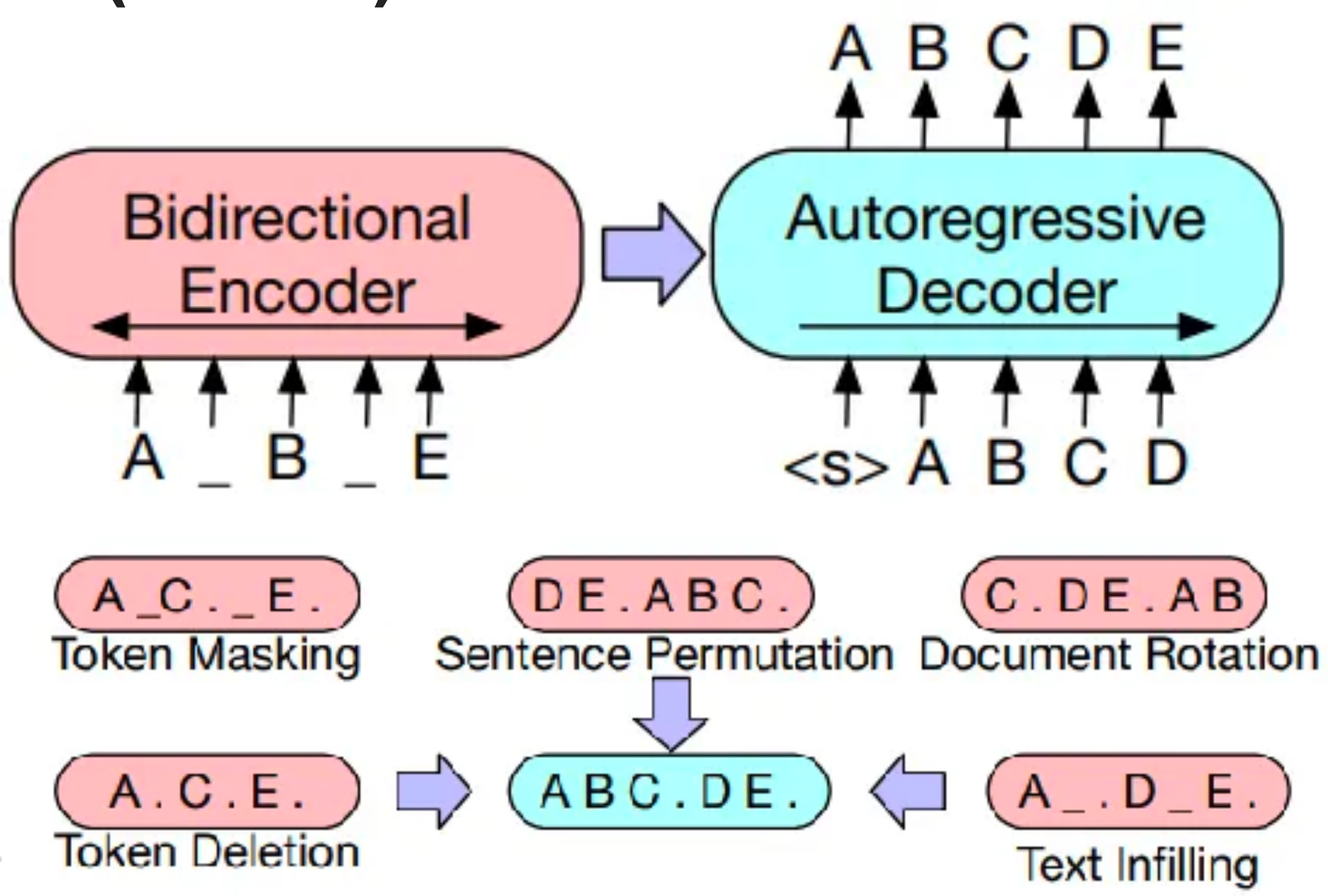
- Earlier Models
- Model T5
- Model BART
- Beam Search

BART

Bidirectional and Auto-Regressive Transformers (BART)

- A bidirectional encoder and an autoregressive decoder.

- BART achieves the state of the art results in the summarization task.



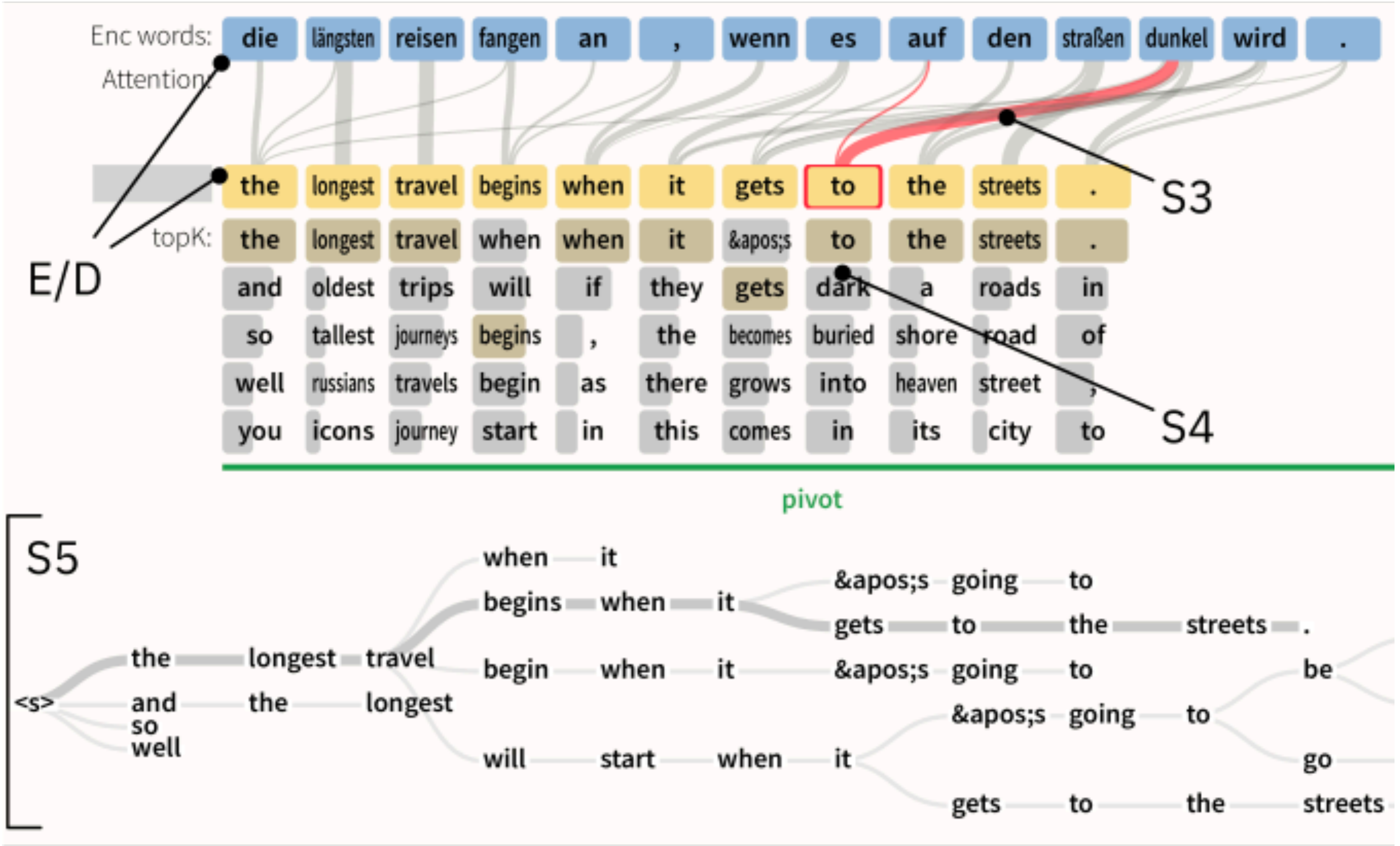
Model	SQuAD 1.1 F1	MNLI Acc	ELI5 PPL	XSum PPL	ConvAI2 PPL	CNN/DM PPL
BERT Base (Devlin et al., 2019)	88.5	**84.3**	-	-	-	-
Masked Language Model	90.0	83.5	24.77	7.87	12.59	7.06
Masked Seq2seq	87.0	82.1	23.40	6.80	11.43	6.19
Language Model	76.7	80.1	**21.40**	7.00	11.51	6.56
Permuted Language Model	89.1	83.7	24.03	7.69	12.23	6.96
Multitask Masked Language Model	89.2	82.4	23.73	7.50	12.39	6.74
BART Base						
w/ Token Masking	90.4	84.1	25.05	7.08	11.73	6.10
w/ Token Deletion	90.4	84.1	24.61	6.90	11.46	5.87
w/ Text Infilling	**90.8**	84.0	24.26	**6.61**	**11.05**	5.83
w/ Document Rotation	77.2	75.3	53.69	17.14	19.87	10.59
w/ Sentence Shuffling	85.4	81.5	41.87	10.93	16.67	7.89
w/ Text Infilling + Sentence Shuffling	**90.8**	83.8	24.17	6.62	11.12	**5.41**

Lewis, Mike, et al.
"Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension." *ACL 2020*.

Encoder–Decoder

- Earlier Models
- Model T5
- Model BART
- **Beam Search**

Beam Search for Decoding



- بخش قابل توجهی از اسلایدها با استفاده از منابع زیرست و یا از آنها الهام گرفته شده است:

- Princeton COS 597G - Understanding Large Language Models
- Stanford CS224 - Deep NLP
- Stanford CS324 - Large Language Models
- Pittsburgh CS1678 - Deep Learning
- UC Berkeley Info256 - NLP

- Dan Jurafsky and James H. Martin. Speech and Language Processing (3rd ed. Draft).

Some Recommended Papers to Read

Transformer Introduction
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Positional Embedding
Dufter, Philipp, Martin Schmitt, and Hinrich Schütze. "Position information in transformers: An overview." Computational Linguistics 48.3 (2022): 733-763.

Yu-An Wang and Yun-Nung Chen. 2020. What Do Position Embeddings Learn? An Empirical Study of Pre-Trained Language Model Positional Encoding. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6840–6849, Online. Association for Computational Linguistics.

Attention Alternatives
Bello, Irwan. "Lambdanetworks: Modeling long-range interactions without attention." arXiv preprint arXiv:2102.08602 (2021).

Zaheer, Manzil, et al. "Big bird: Transformers for longer sequences." Advances in neural information processing systems 33 (2020): 17283-17297.

Subword Tokenization Issues in Transformers
Kaj Bostrom and Greg Durrett. 2020. Byte Pair Encoding is Suboptimal for Language Model Pretraining. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4617–4624, Online. Association for Computational Linguistics.

A Deep Dive into BERT model architecture
Rogers, Anna, Olga Kovaleva, and Anna Rumshisky. "A primer in BERTology: What we know about how BERT works." Transactions of the Association for Computational Linguistics 8 (2021): 842-866.

GPT Model Few Shot Learning
Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

T5 Model
Raffel, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." The Journal of Machine Learning Research 21.1 (2020): 5485-5551.

Visual Transformers
Y. Liu et al., "A Survey of Visual Transformers," in IEEE Transactions on Neural Networks and Learning Systems, doi: 10.1109/TNNLS.2022.3227717.