بسم الله الرحمن الرحیم

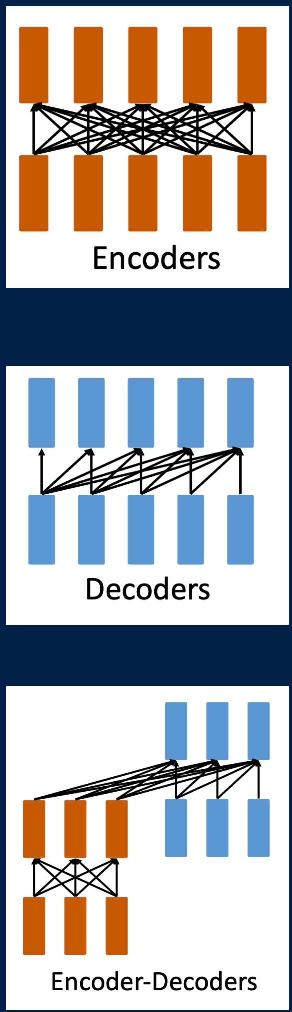
Transformer Language models

Lecture 3 - Transformers (ii)

Oct. 10th 2023

Artificial Intelligence Group
Computer Engineering Department, SUT

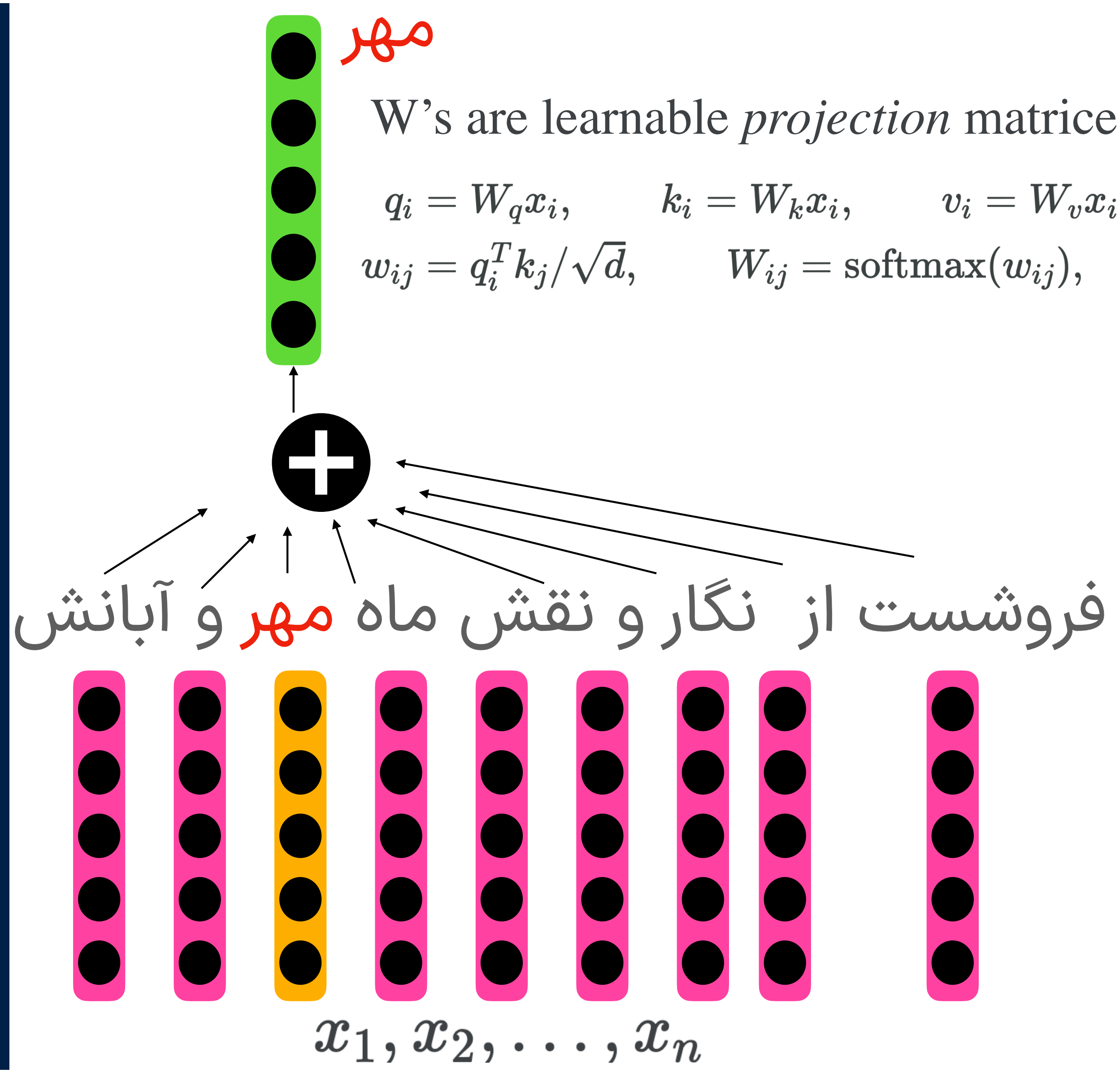
Transformer Architectures



- 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

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

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

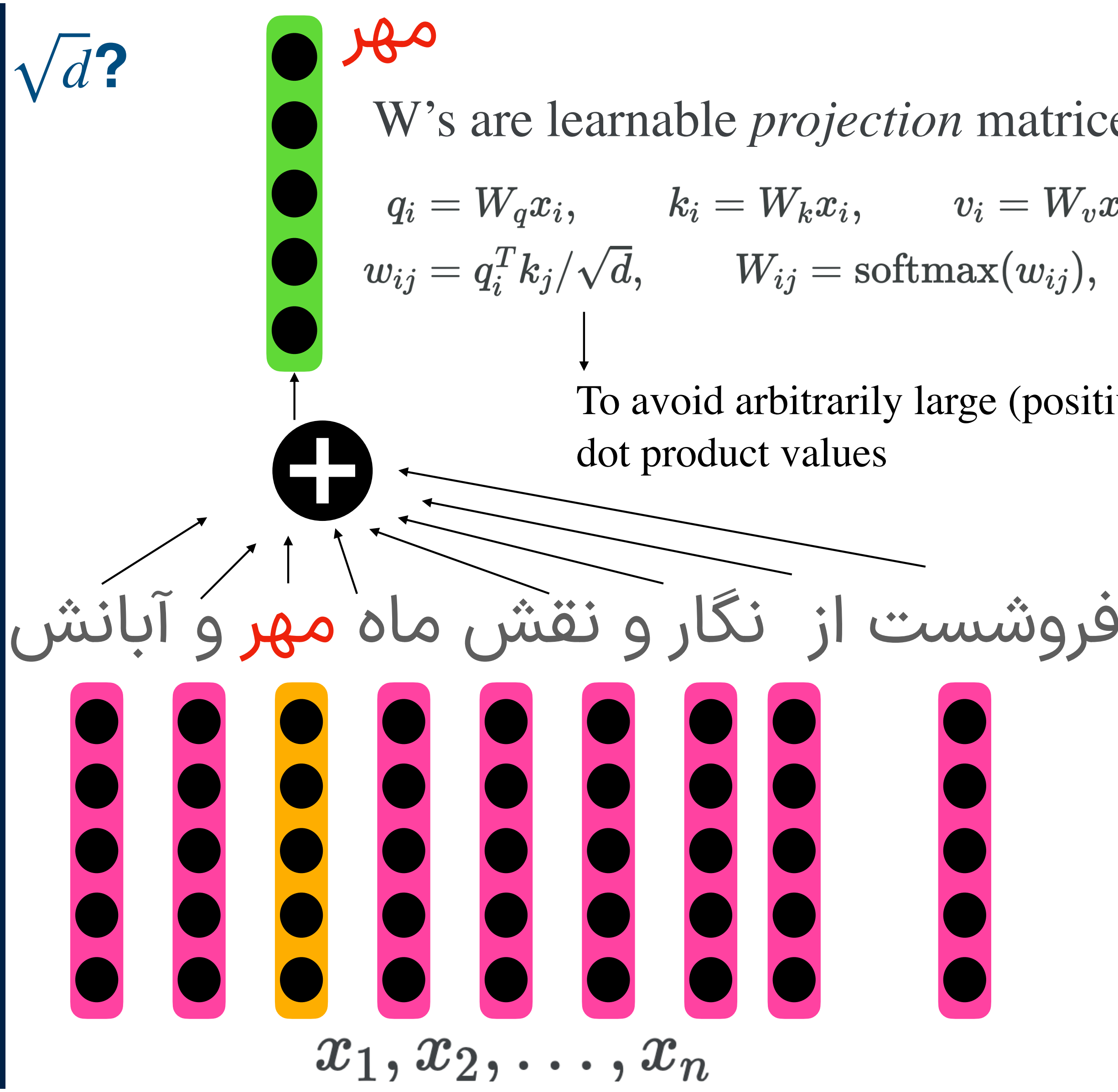
Why scaling by $\sqrt{d}$?

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 $\sqrt{d}$?

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

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

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

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

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

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

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

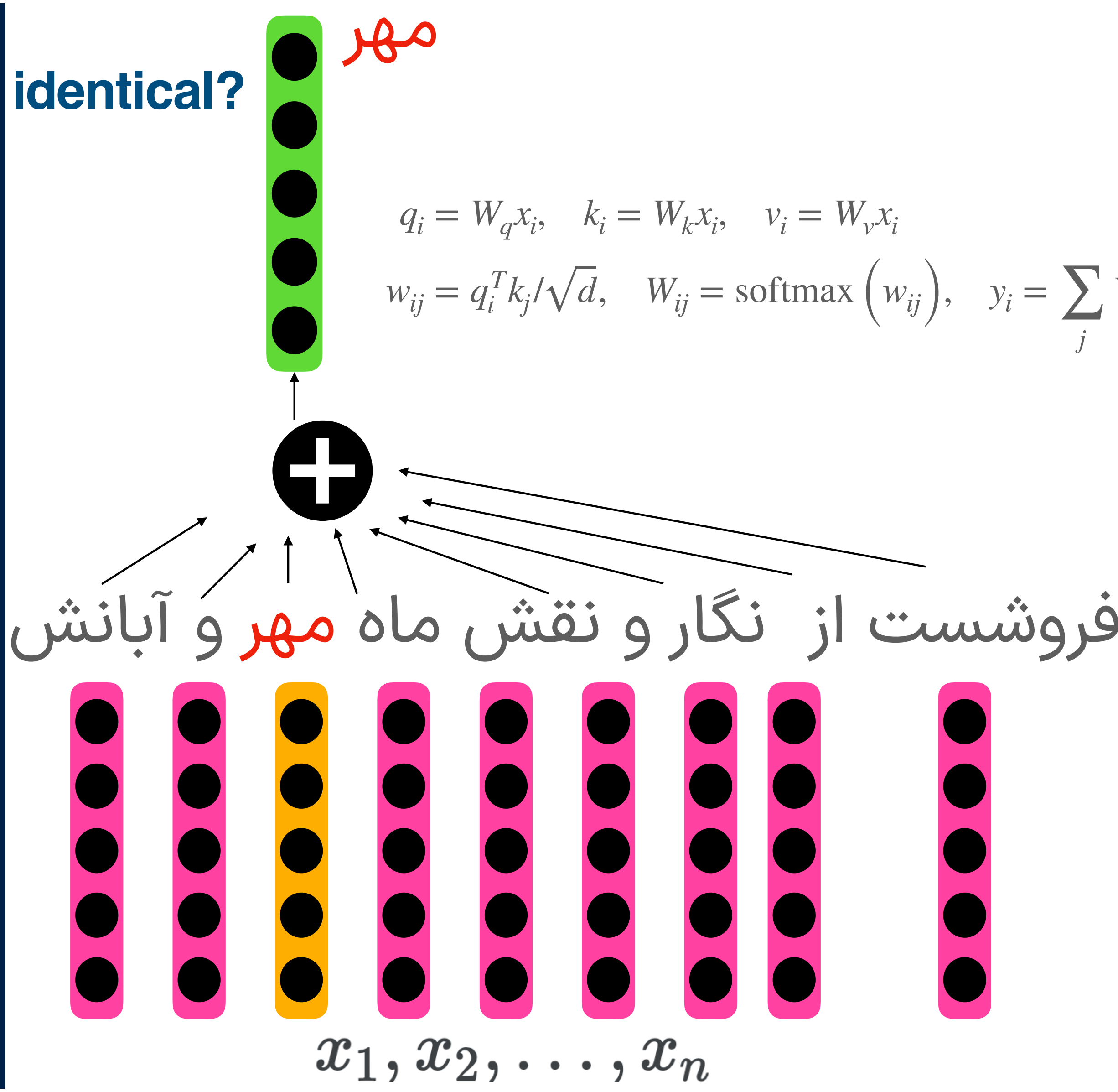
$$w_{ij} = \frac{q_i^T k_j - \mu}{\sigma} = \frac{q_i^T k_j}{\sqrt{d}}$$

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

Are W_k and W_q identical?

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

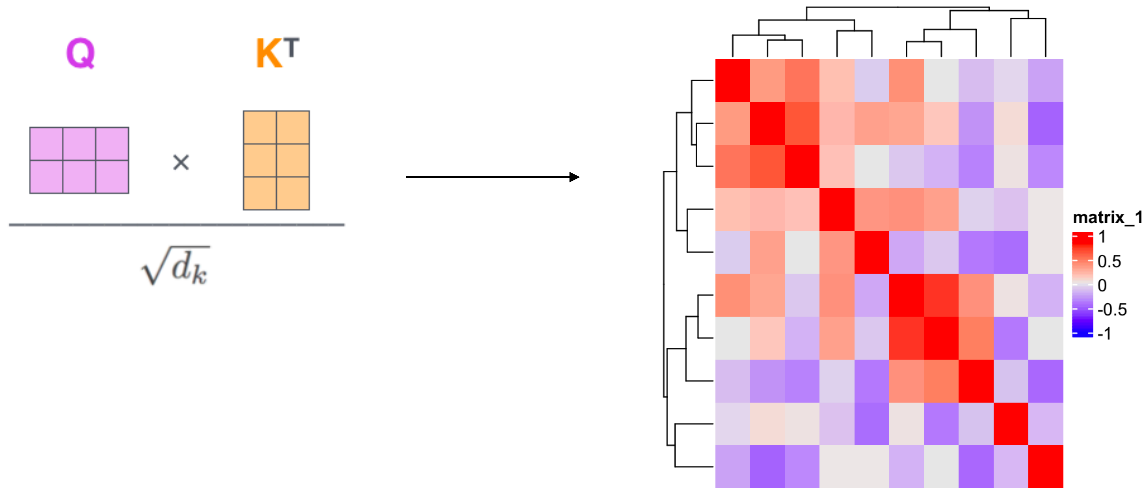
$$w_{ij} = q_i^T k_j / \sqrt{d}, \quad W_{ij} = \mathrm{softmax}\left(w_{ij}\right), \quad y_i = \sum_j W_{ij} v_j \,.$$



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

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

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



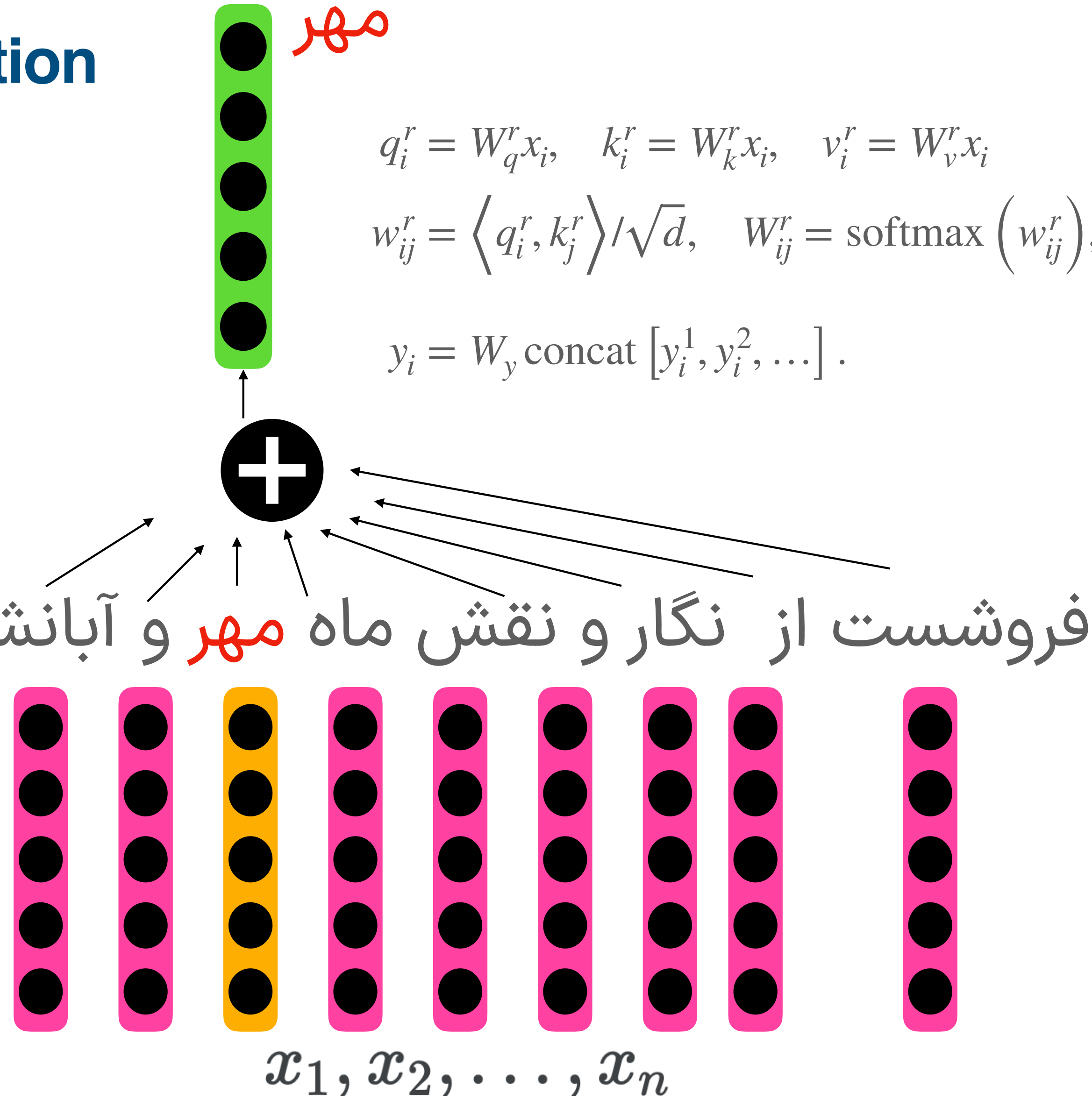
Multihead Attention

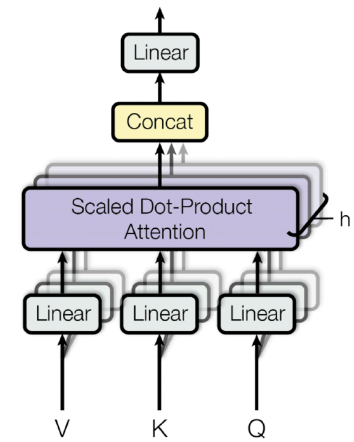
REVIEW

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

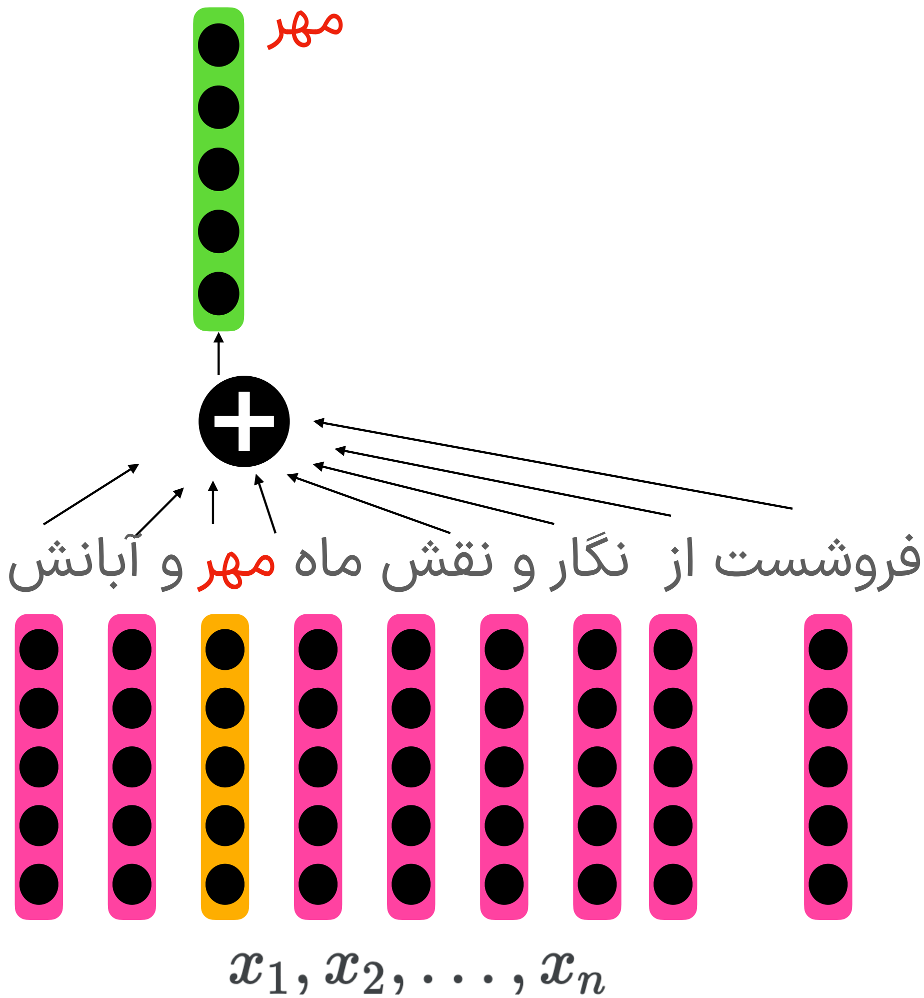
$$w_{ij}^r = \left\langle q_i^r, k_j^r \right\rangle / \sqrt{d}, \quad W_{ij}^r = \mathrm{softmax}\left(w_{ij}^r\right), \quad y_i^r = \sum_j W_{ij}^r v_j,$$

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





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

مهرماه باران می بارد

معمولا هرسال مهرماه باران می بارد

Positional Embedding types?

ABSOLUTE VS. RELATIVE POSITION ENCODING

مهر ماه باران می بارد

معمولا هر سال مهر ماه باران می بارد

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

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

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

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

Modifying Attention Matrix

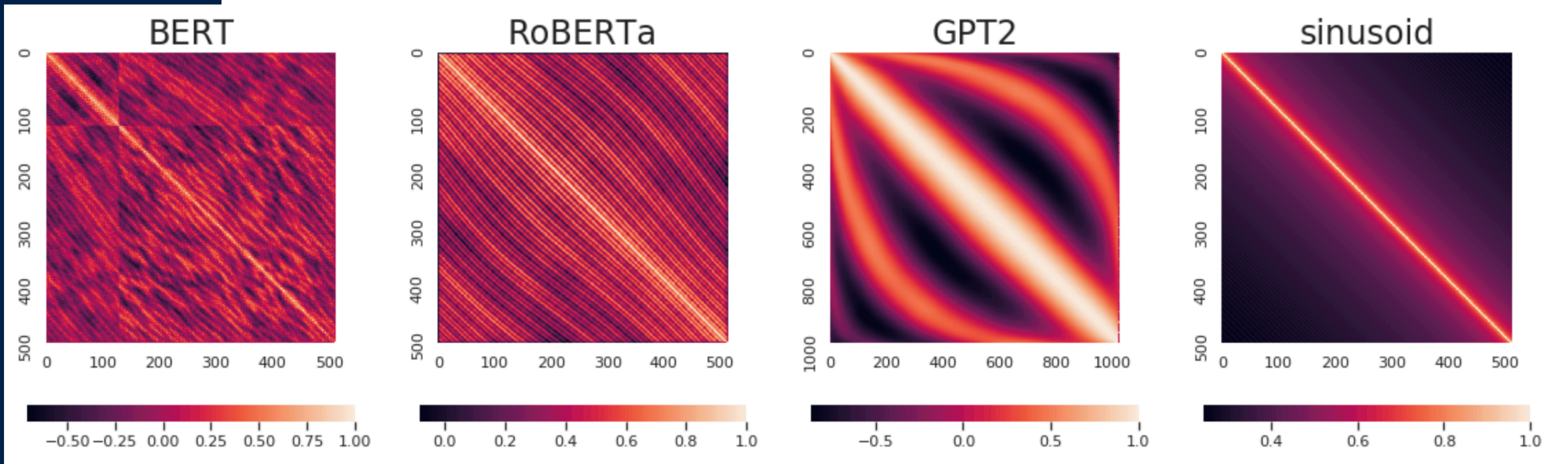
Input Embedding $U \in \mathbb{R}^{\times d}$

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.

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

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

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

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

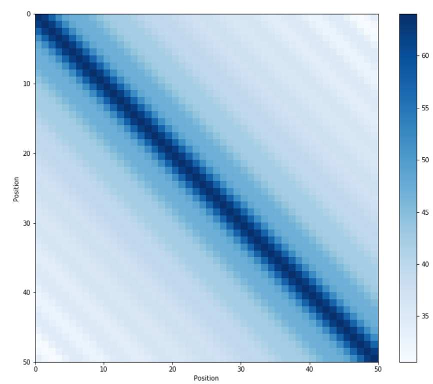
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

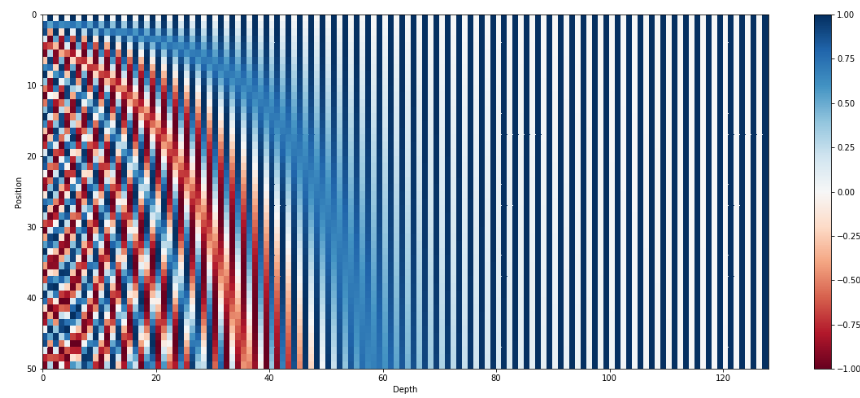
Sinusoidal Positional Embedding

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

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.

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





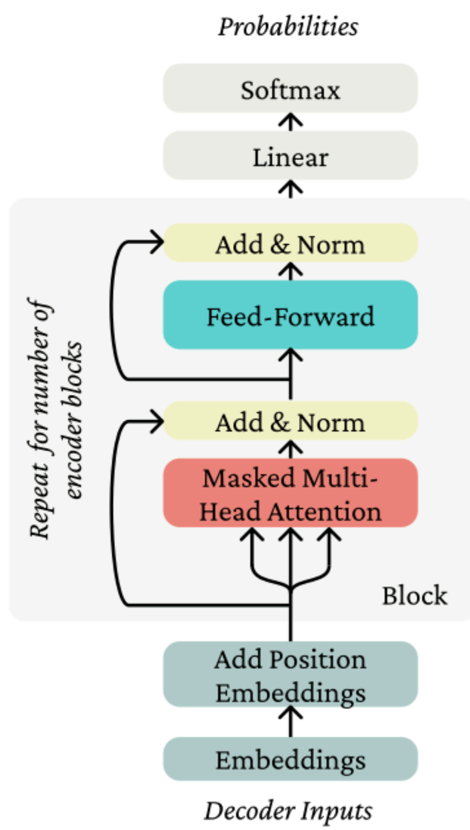
Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

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   Layer norm

d-dimensional vectors for each sample in batch

$a_1, a_2, \ldots, a_B$

d-dim

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

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

different *dimensions* of a

1-dim

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

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

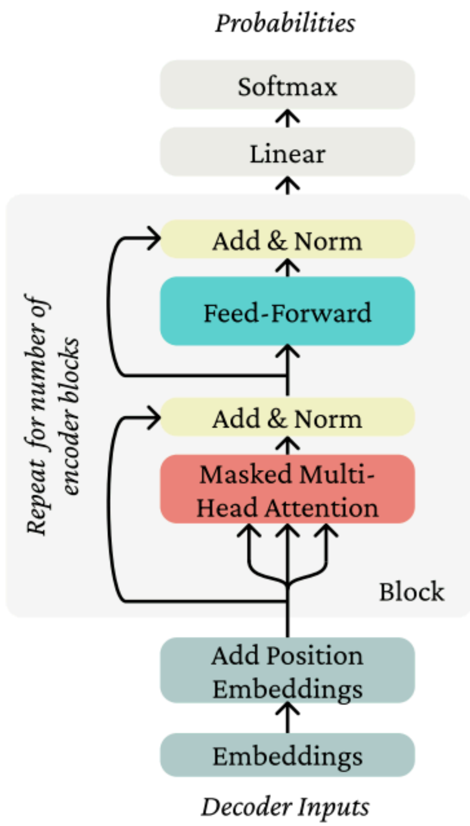
Why transformers?

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

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.

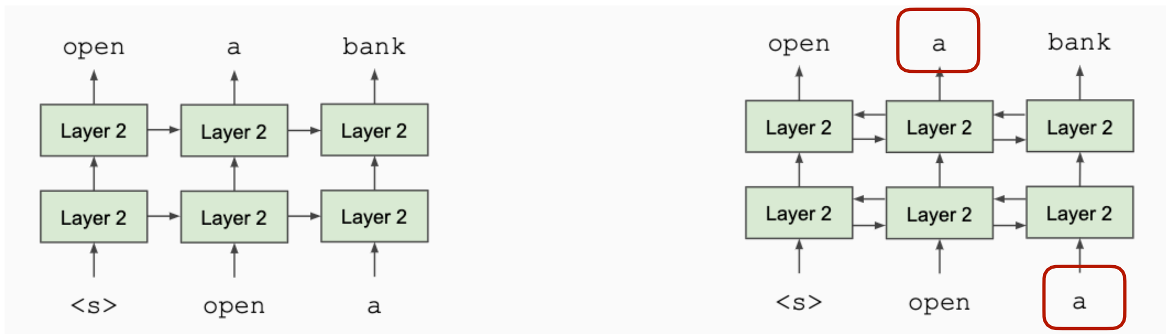
Example #1: we went to the river bank.

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

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

MLM: masking rate and strategy

- **Q: What is the value of *k*?**
  - They always use k = 15%.
  - Too little masking: computationally expensive
  - 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.