Attention & Transformer

CS XXX: Introduction to Large Language Models

Attention Intuition

- Intuition: a representation of meaning of a word should be different in different contexts!

`The` **`chicken`** `didn't cross the road because it was too tired`

`The chicken didn't cross the` **`road`** `because it was too wide`

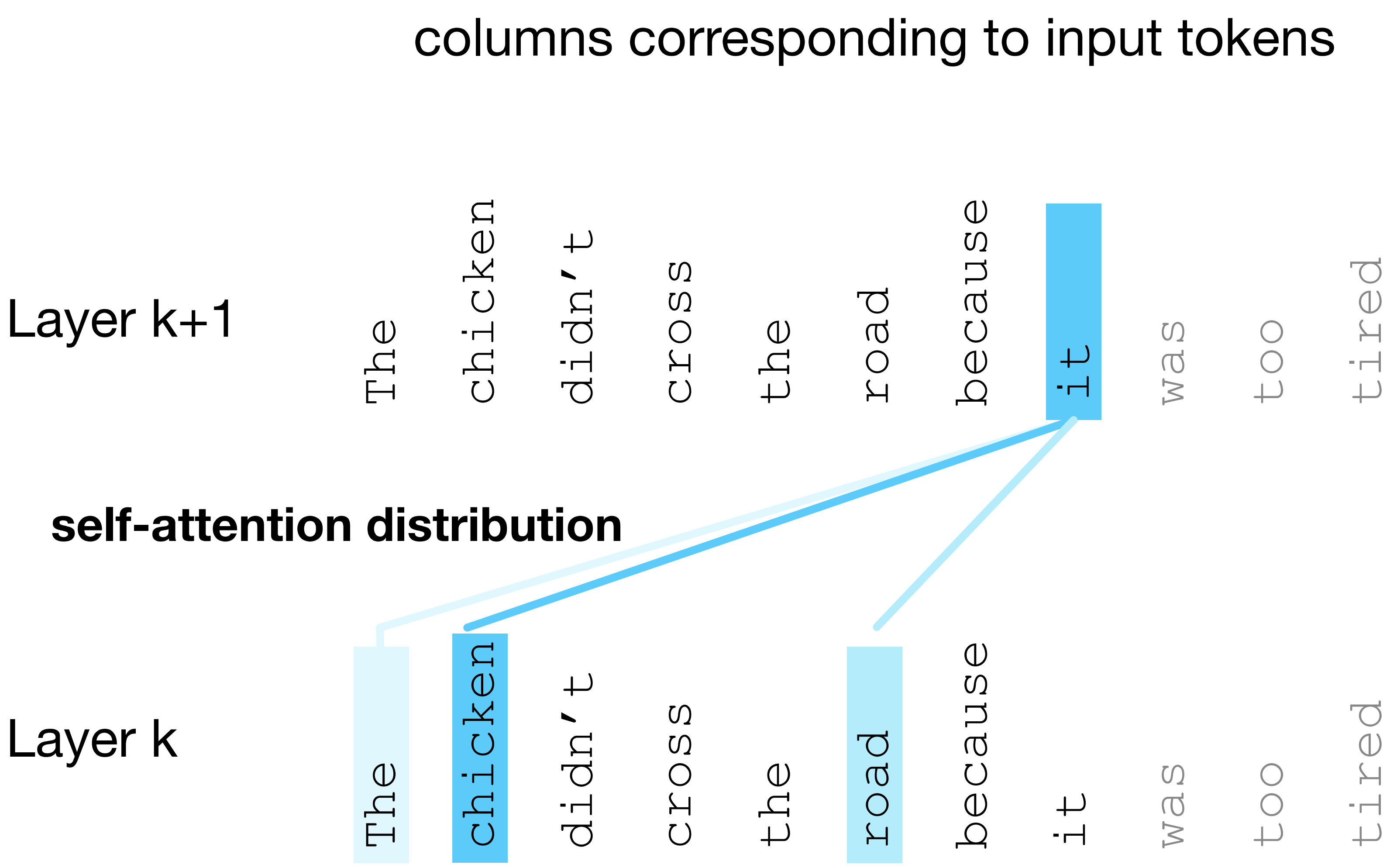
- “it” has a different meaning in different contexts

Attention Intuition

- Intuition: a representation of meaning of a word should be different in different contexts!
- Contextual Embedding: each word has a different vector that expresses different meanings depending on the surrounding words
- How to compute contextual embeddings?
  - **Attention**

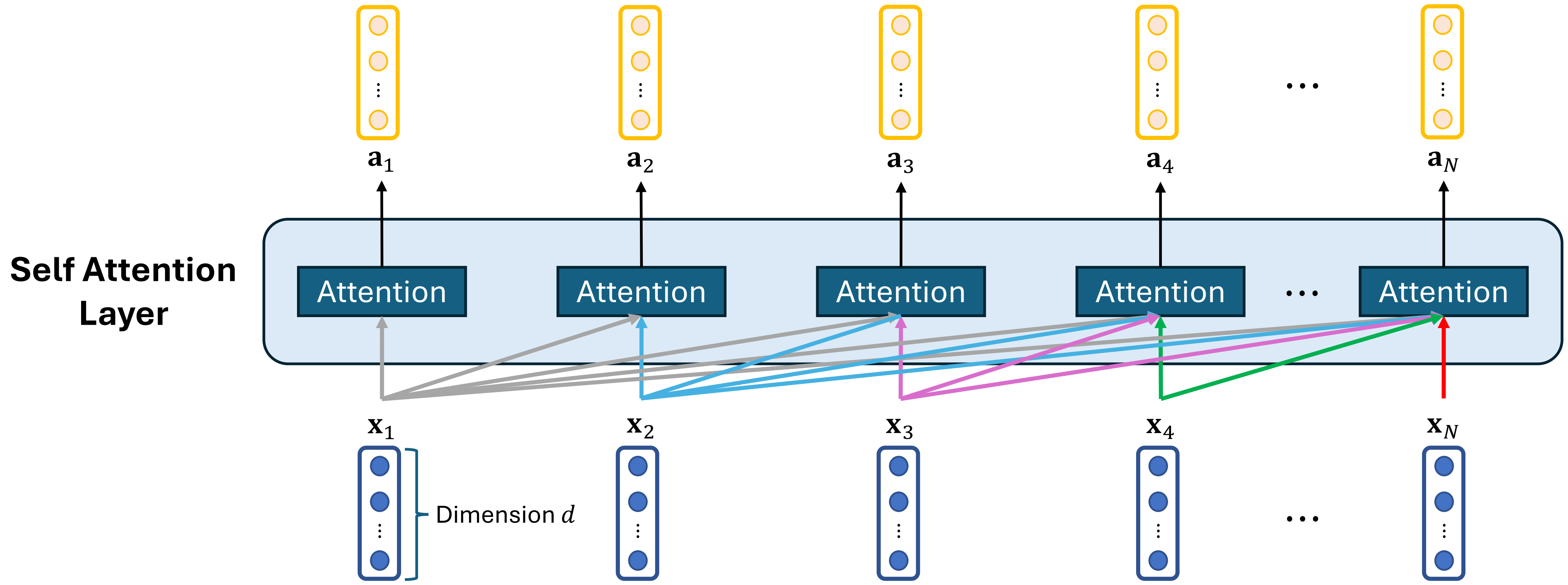
Attention Intuition

- Attention is comparison of input to other input elements.



Attention Intuition

- A mechanism for helping compute the embedding for a token by selectively attending to and integrating information from surrounding tokens (at the previous layer).



Attention Intuition

- Self Attention (Simplified)
  - Given a sequence of token embeddings:

$$\mathbf{x}_1 \quad \mathbf{x}_2 \quad \mathbf{x}_3 \quad \mathbf{x}_4 \quad \mathbf{x}_5 \quad \mathbf{x}_6 \quad \mathbf{x}_7 \quad \mathbf{x}_i$$

  - Produce: $\mathbf{a}_i$ = a weighted sum of $\mathbf{x}_1$ through $\mathbf{x}_7$ (and $\mathbf{x}_i$) Weighted by their similarity to $\mathbf{x}_i$

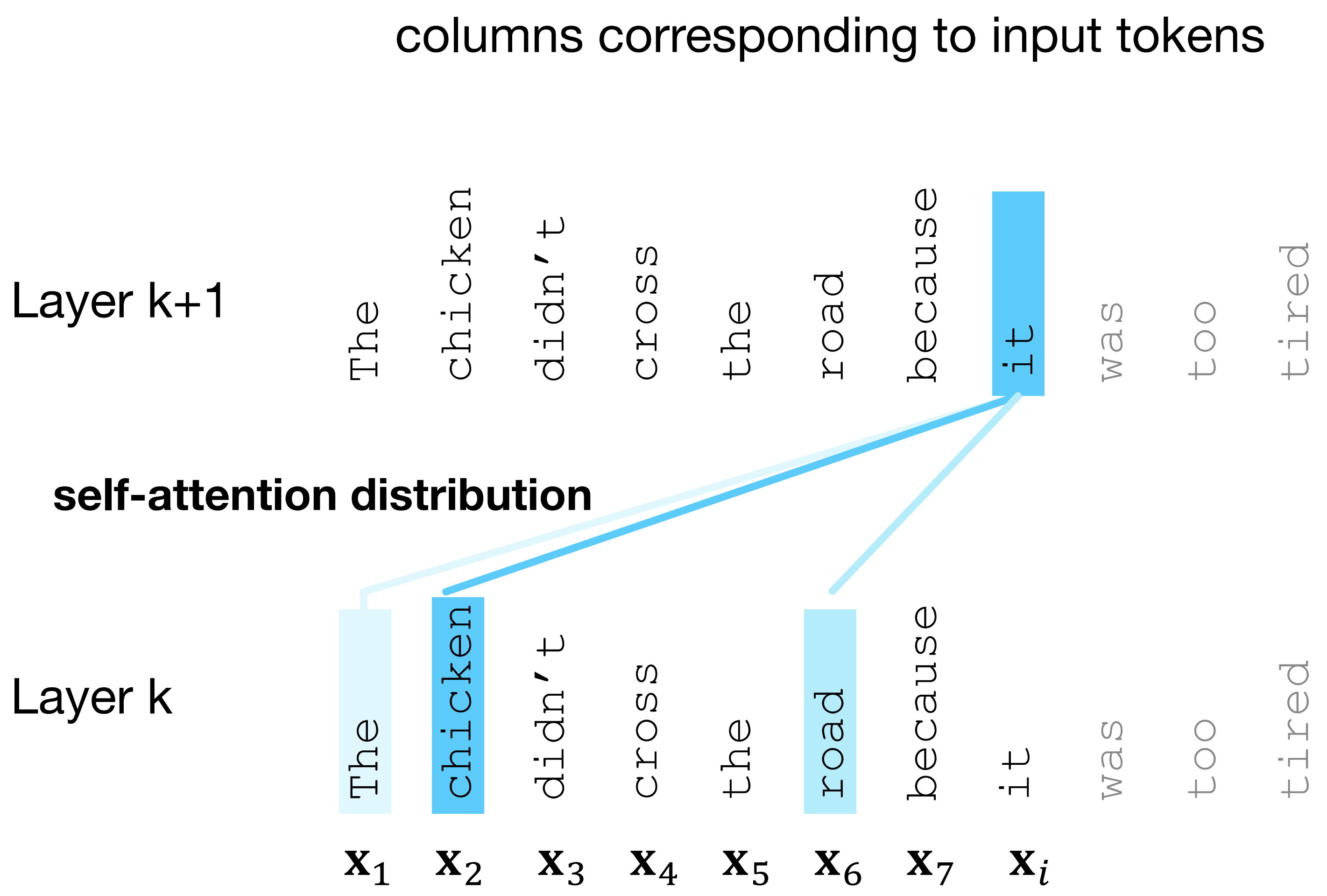
$$\mathrm{Score}_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j, \qquad \forall j \leq i$$

$$\alpha_{ij} = \mathrm{Softmax}\left(\mathrm{Score}_{ij}\right), \qquad \forall j \leq i$$

$$\mathbf{a}_i = \sum_{j \leq i} \alpha_{ij} \mathbf{x}_j$$

Attention Intuition

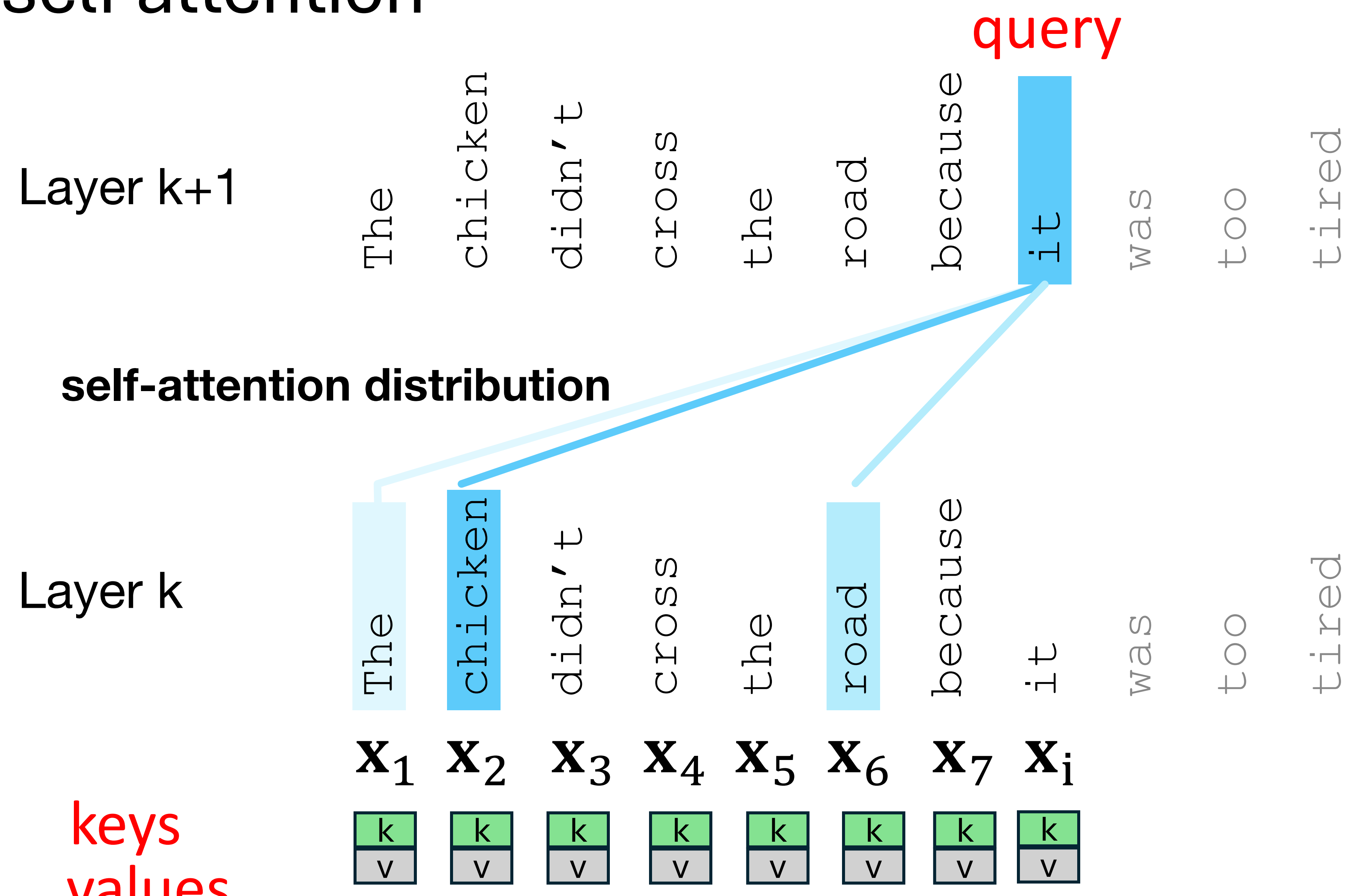
- Self Attention (Simplified)



Actual self attention

- An Actual Attention Head: slightly more complicated
- High-level idea: instead of using vectors (like $\mathbf{x}_4$) directly, we'll represent $\mathbf{x}_i$ in 3 separate roles (projection vectors of $\mathbf{x}_i$):
  - **query:** as the current element being compared to the preceding inputs.
  - **key:** as a preceding input that is being compared to the current element to determine a similarity
  - **value:** a value of a preceding element that gets weighted and summed

Actual self attention



Actual self attention

- We'll use matrices to project each vector $\mathbf{x}_i$ into a representation of its role as query, key, value:
    - **query**: $\mathbf{W}^{\mathrm{Q}} \in \mathbb{R}^{d \times d_k}$
    - **key**: $\mathbf{W}^{\mathrm{K}} \in \mathbb{R}^{d \times d_k}$
    - **value**: $\mathbf{W}^{\mathrm{V}} \in \mathbb{R}^{d \times d_v}$

$$\mathbf{q}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{Q}} \qquad \mathbf{k}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{k}} \qquad \mathbf{v}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{V}}$$

Actual self attention

- Given these 3 representation of $\mathbf{x}_i$

$$\mathbf{q}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{Q}} \qquad \mathbf{k}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{k}} \qquad \mathbf{v}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{V}}$$

- To compute similarity of current element $\mathbf{x}_i$ with some prior element $\mathbf{x}_j$
    - We’ll use dot product between $\mathbf{q}_i$ and $\mathbf{k}_j$.
    - And instead of summing up $\mathbf{x}_j$, we'll sum up $\mathbf{v}_j$

Actual self attention

$$\mathbf{q}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{Q}} \qquad \mathbf{k}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{k}} \qquad \mathbf{v}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{V}}$$

$$\mathrm{score}_{ij} = \frac{\mathbf{q}_i \cdot \mathbf{k}_j}{\sqrt{d_k}}, \qquad \forall j \leq i$$

$$\alpha_{ij} = \mathrm{Softmax}\left(\mathrm{Score}_{ij}\right), \qquad \forall j \leq i$$

$$\mathbf{a}_i = \sum_{j \leq i} \alpha_{ij}\mathbf{v}_j$$

Actual self attention

Visual look of calculating $\mathbf{a}_3$

$$\mathbf{q}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{Q}}$$

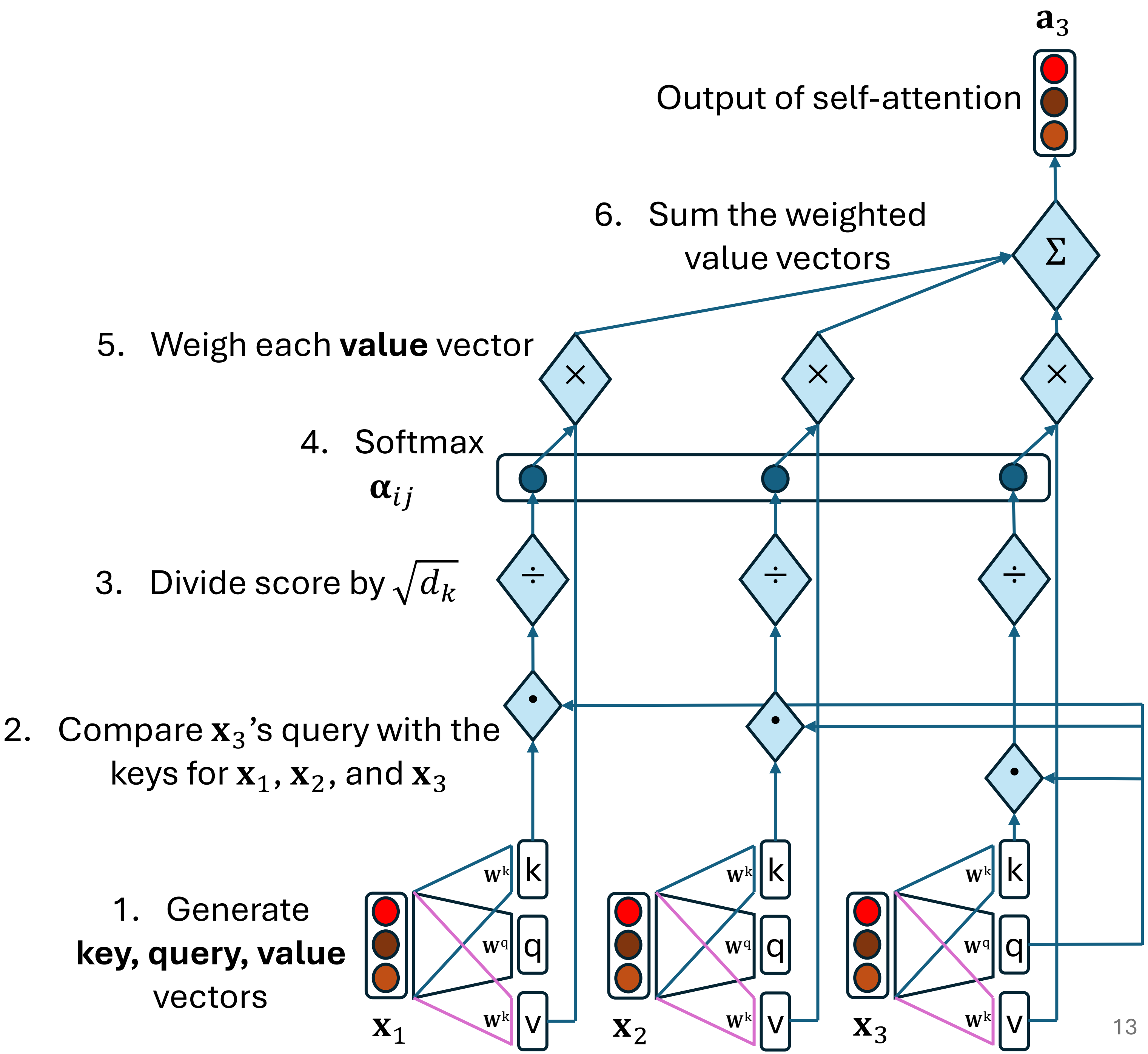
$$\mathbf{k}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{k}}$$

$$\mathbf{v}_{\mathrm{i}} = \mathbf{x}_{\mathrm{i}}\mathbf{W}^{\mathrm{V}}$$

$$\mathrm{score}_{ij} = \frac{\mathbf{q}_i \cdot \mathbf{k}_j}{\sqrt{d_k}}, \qquad \forall j \leq i$$

$$\alpha_{ij} = \mathrm{Softmax}(\mathrm{Score}_{ij}),\ \forall j \leq i$$

$$\mathbf{a}_i = \sum_{j \leq i} \alpha_{ij}\mathbf{v}_j$$

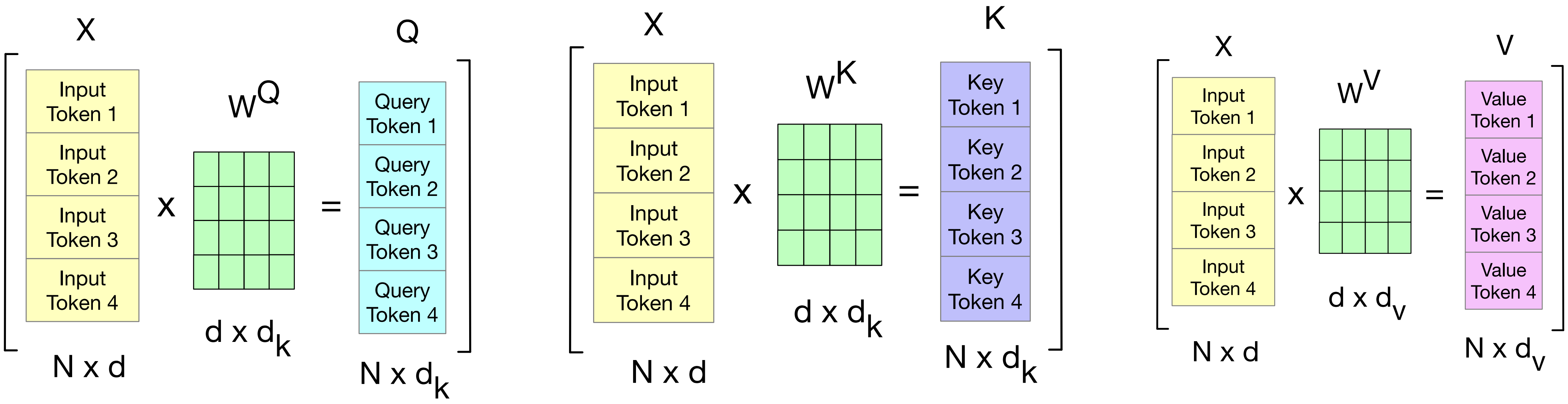


Actual self attention

Parallelizing Computation using Input Matrix $\mathbf{X}$

- We can pack the N tokens of the input sequence into a single matrix $\mathbf{X}$ of size $[N \ \times \ d]$.
- Each row of $\mathbf{X}$ is the embedding of one token of the input.
- $\mathbf{X}$ can have 1K - 32K rows, each of the dimensionality of the embedding d (the **model dimension**)

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{\mathbf{Q}}; \ \ \mathbf{K} = \mathbf{X}\mathbf{W}^{\mathbf{K}}; \ \ \mathbf{V} = \mathbf{X}\mathbf{W}^{\mathbf{V}}$$



Actual self attention

Parallelizing Computation using Input Matrix $\mathbf{X}$

$$\text{Score } (\boldsymbol{S}) = \frac{\mathbf{QK^T}}{\sqrt{d_k}}$$

$$\underset{N \times d_k}{\overset{Q}{\begin{bmatrix} q1 \\ q2 \\ q3 \\ q4 \end{bmatrix}}} \times \underset{d_k \times N}{\overset{K^T}{\begin{bmatrix} k1 & k2 & k3 & k4 \end{bmatrix}}} = \underset{N \times N}{\overset{QK^T}{\begin{bmatrix} q1 \cdot k1 & q1 \cdot k2 & q1 \cdot k3 & q1 \cdot k4 \\ q2 \cdot k1 & q2 \cdot k2 & q2 \cdot k3 & q2 \cdot k4 \\ q3 \cdot k1 & q3 \cdot k2 & q3 \cdot k3 & q3 \cdot k4 \\ q4 \cdot k1 & q4 \cdot k2 & q4 \cdot k3 & q4 \cdot k4 \end{bmatrix}}} \xrightarrow{\frac{1}{\sqrt{d_k}}} \underset{N \times N}{\overset{\mathbf{S} = \frac{\mathbf{QK^T}}{\sqrt{d_k}}}{\begin{bmatrix} \frac{\mathbf{q1 \cdot k1}}{\sqrt{d_k}} & \frac{\mathbf{q1 \cdot k2}}{\sqrt{d_k}} & \frac{\mathbf{q1 \cdot k3}}{\sqrt{d_k}} & \frac{\mathbf{q1 \cdot k4}}{\sqrt{d_k}} \\ \frac{\mathbf{q2 \cdot k1}}{\sqrt{d_k}} & \frac{\mathbf{q2 \cdot k2}}{\sqrt{d_k}} & \frac{\mathbf{q2 \cdot k3}}{\sqrt{d_k}} & \frac{\mathbf{q2 \cdot k4}}{\sqrt{d_k}} \\ \frac{\mathbf{q3 \cdot k1}}{\sqrt{d_k}} & \frac{\mathbf{q3 \cdot k2}}{\sqrt{d_k}} & \frac{\mathbf{q3 \cdot k3}}{\sqrt{d_k}} & \frac{\mathbf{q3 \cdot k4}}{\sqrt{d_k}} \\ \frac{\mathbf{q4 \cdot k1}}{\sqrt{d_k}} & \frac{\mathbf{q4 \cdot k2}}{\sqrt{d_k}} & \frac{\mathbf{q4 \cdot k3}}{\sqrt{d_k}} & \frac{\mathbf{q4 \cdot k4}}{\sqrt{d_k}} \end{bmatrix}}}$$

Actual self attention

Parallelizing Computation using Input Matrix $\mathbf{X}$

- **Attention score** $(\boldsymbol{\alpha})=$ $\mathbf{softmax}\left(\frac{\mathbf{QK^T}}{\sqrt{d_k}}\right)$

$$\text{softmax}\left[\begin{array}{c}\frac{\mathbf{QK^T}}{\sqrt{d_k}} \\ \begin{bmatrix} \frac{\mathbf{q}1\cdot\mathbf{k}1}{\sqrt{d_k}} & \frac{\mathbf{q}1\cdot\mathbf{k}2}{\sqrt{d_k}} & \frac{\mathbf{q}1\cdot\mathbf{k}3}{\sqrt{d_k}} & \frac{\mathbf{q}1\cdot\mathbf{k}4}{\sqrt{d_k}} \\ \frac{\mathbf{q}2\cdot\mathbf{k}1}{\sqrt{d_k}} & \frac{\mathbf{q}2\cdot\mathbf{k}2}{\sqrt{d_k}} & \frac{\mathbf{q}2\cdot\mathbf{k}3}{\sqrt{d_k}} & \frac{\mathbf{q}2\cdot\mathbf{k}4}{\sqrt{d_k}} \\ \frac{\mathbf{q}3\cdot\mathbf{k}1}{\sqrt{d_k}} & \frac{\mathbf{q}3\cdot\mathbf{k}2}{\sqrt{d_k}} & \frac{\mathbf{q}3\cdot\mathbf{k}3}{\sqrt{d_k}} & \frac{\mathbf{q}3\cdot\mathbf{k}4}{\sqrt{d_k}} \\ \frac{\mathbf{q}4\cdot\mathbf{k}1}{\sqrt{d_k}} & \frac{\mathbf{q}4\cdot\mathbf{k}2}{\sqrt{d_k}} & \frac{\mathbf{q}4\cdot\mathbf{k}3}{\sqrt{d_k}} & \frac{\mathbf{q}4\cdot\mathbf{k}4}{\sqrt{d_k}} \end{bmatrix} \\ \text{N x N}\end{array}\right] = \begin{array}{c}\boldsymbol{\alpha} \\ \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \alpha_{34} \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{bmatrix} \\ \text{N x N}\end{array}$$

Actual self attention

Parallelizing Computation using Input Matrix $\mathbf{X}$

- An attention vector for each input token

$$\textbf{Self–Attention (A)} = \left(\text{softmax}\left(\frac{\mathbf{QK^T}}{\sqrt{d_k}}\right)\right)\mathbf{V}$$

$$\overset{\boldsymbol{\alpha}}{\underset{\text{N x N}}{\begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & \alpha_{34} \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{bmatrix}}} \times \overset{\text{V}}{\underset{\text{N x } d_V}{\begin{bmatrix} \textbf{v1} \\ \textbf{v2} \\ \textbf{v3} \\ \textbf{v4} \end{bmatrix}}} = \overset{\text{A}}{\underset{\text{N x } d_V}{\begin{bmatrix} \textbf{a1} \\ \textbf{a2} \\ \textbf{a3} \\ \textbf{a4} \end{bmatrix}}}$$

Actual self attention

Masking the future: Masked Self-Attention

$$\textbf{Self–Attention (A)} = \left(\text{softmax} \left(\overbrace{\textbf{mask}\left(\frac{\mathbf{QK^T}}{\sqrt{d_k}} \right)} \right) \right) \mathbf{V}$$

- Add $-\infty$ to cells in upper triangle
- The softmax will turn it to 0

$$\text{mask}\left[\begin{array}{c} \frac{\mathbf{QK^T}}{\sqrt{d_k}} \\ \begin{bmatrix} \frac{\mathbf{q1}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q1}\cdot\mathbf{k2}}{\sqrt{d_k}} & \frac{\mathbf{q1}\cdot\mathbf{k3}}{\sqrt{d_k}} & \frac{\mathbf{q1}\cdot\mathbf{k4}}{\sqrt{d_k}} \\ \frac{\mathbf{q2}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q2}\cdot\mathbf{k2}}{\sqrt{d_k}} & \frac{\mathbf{q2}\cdot\mathbf{k3}}{\sqrt{d_k}} & \frac{\mathbf{q2}\cdot\mathbf{k4}}{\sqrt{d_k}} \\ \frac{\mathbf{q3}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q3}\cdot\mathbf{k2}}{\sqrt{d_k}} & \frac{\mathbf{q3}\cdot\mathbf{k3}}{\sqrt{d_k}} & \frac{\mathbf{q3}\cdot\mathbf{k4}}{\sqrt{d_k}} \\ \frac{\mathbf{q4}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q4}\cdot\mathbf{k2}}{\sqrt{d_k}} & \frac{\mathbf{q4}\cdot\mathbf{k3}}{\sqrt{d_k}} & \frac{\mathbf{q4}\cdot\mathbf{k4}}{\sqrt{d_k}} \end{bmatrix} \\ \text{N x N} \end{array} \right] = \begin{array}{c} \frac{\mathbf{QK^T}}{\sqrt{d_k}} \\ \begin{bmatrix} \frac{\mathbf{q1}\cdot\mathbf{k1}}{\sqrt{d_k}} & -\infty & -\infty & -\infty \\ \frac{\mathbf{q2}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q2}\cdot\mathbf{k2}}{\sqrt{d_k}} & -\infty & -\infty \\ \frac{\mathbf{q3}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q3}\cdot\mathbf{k2}}{\sqrt{d_k}} & \frac{\mathbf{q3}\cdot\mathbf{k3}}{\sqrt{d_k}} & -\infty \\ \frac{\mathbf{q4}\cdot\mathbf{k1}}{\sqrt{d_k}} & \frac{\mathbf{q4}\cdot\mathbf{k2}}{\sqrt{d_k}} & \frac{\mathbf{q4}\cdot\mathbf{k3}}{\sqrt{d_k}} & \frac{\mathbf{q4}\cdot\mathbf{k4}}{\sqrt{d_k}} \end{bmatrix} \\ \text{N x N} \end{array}$$

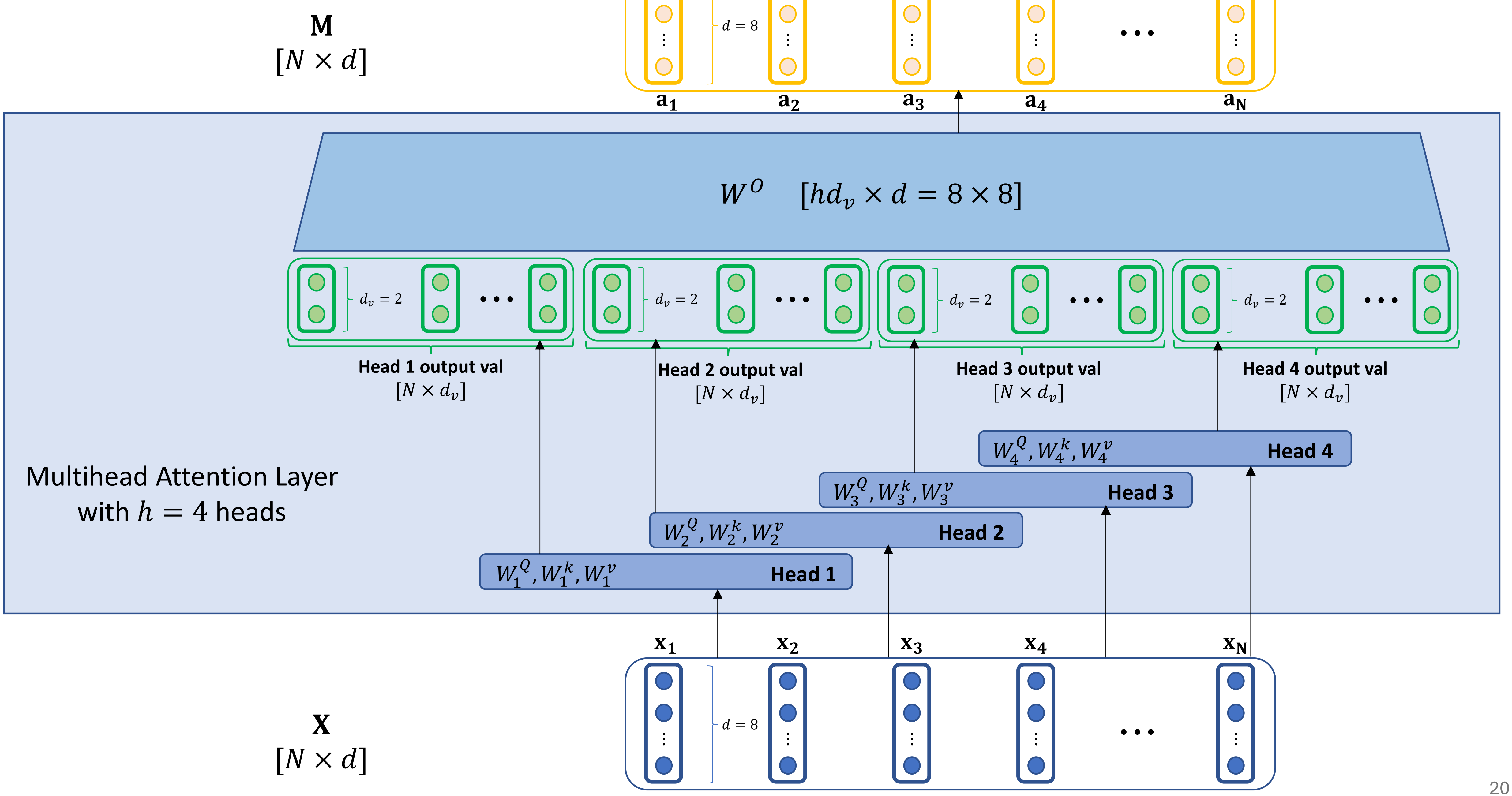
Multi-head Self Attention

$$\mathbf{Q}^{i} = \mathbf{X}\mathbf{W}^{\mathbf{Q}i} \qquad \mathbf{K}^{i} = \mathbf{X}\mathbf{W}^{\mathbf{K}i} \qquad \mathbf{V}^{i} = \mathbf{X}\mathbf{W}^{\mathbf{V}i}$$

$$\text{head}_{i} = \text{Self}-\text{Attention}\ (\mathbf{A}) = \left(\text{softmax}\left(\frac{\mathbf{Q}^{\mathrm{i}}\mathbf{K}^{\mathbf{T}}}{\sqrt{d_k}} \right) \right) \mathbf{V}$$

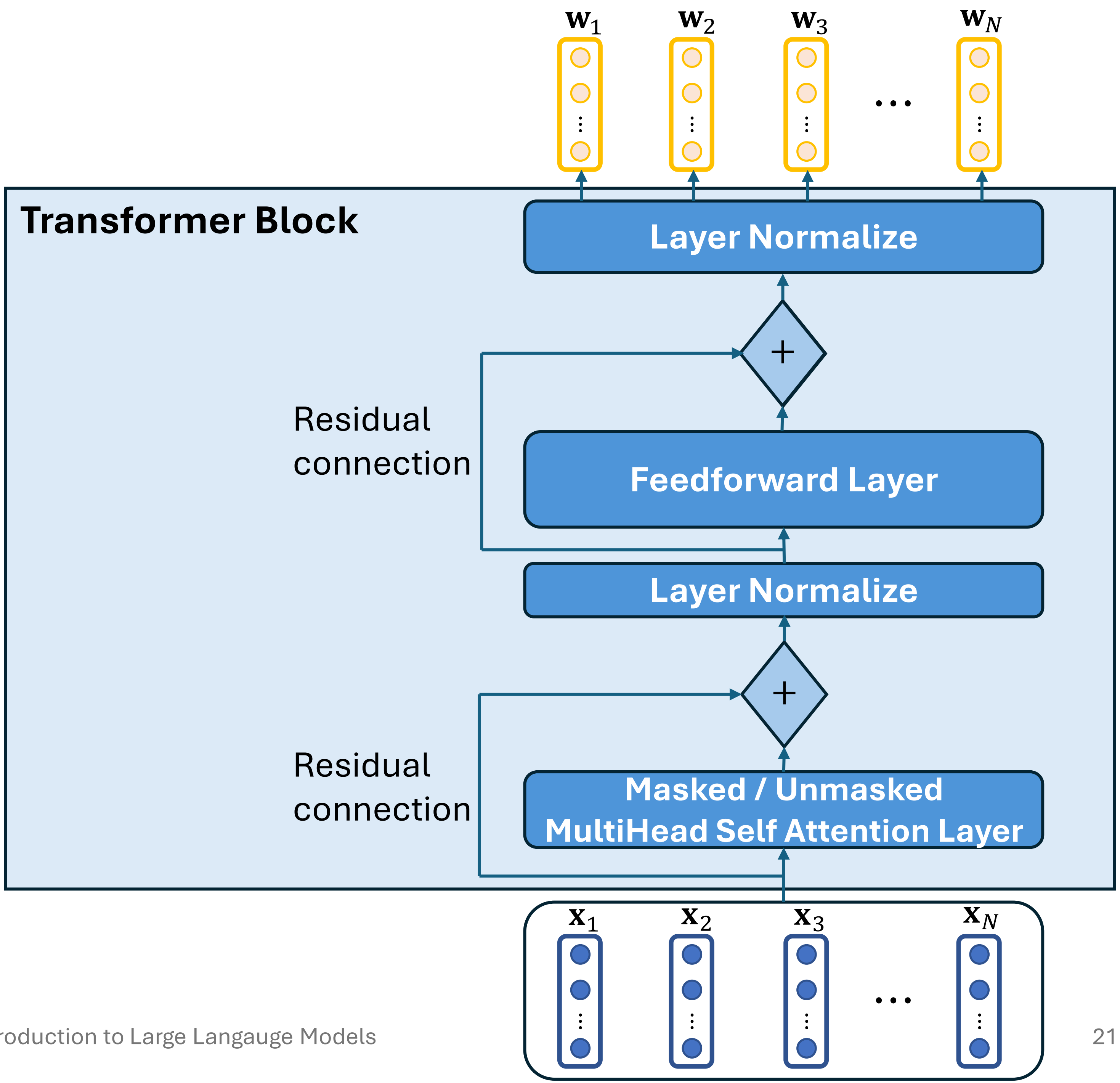
$$\text{MultiHead Attention}\ (\mathbf{M}) = (\text{head}_1 \oplus \text{head}_2 \ \ldots \oplus \text{head}_h)\mathbf{W}^{0}$$

Multi-head Self Attention



Transformer

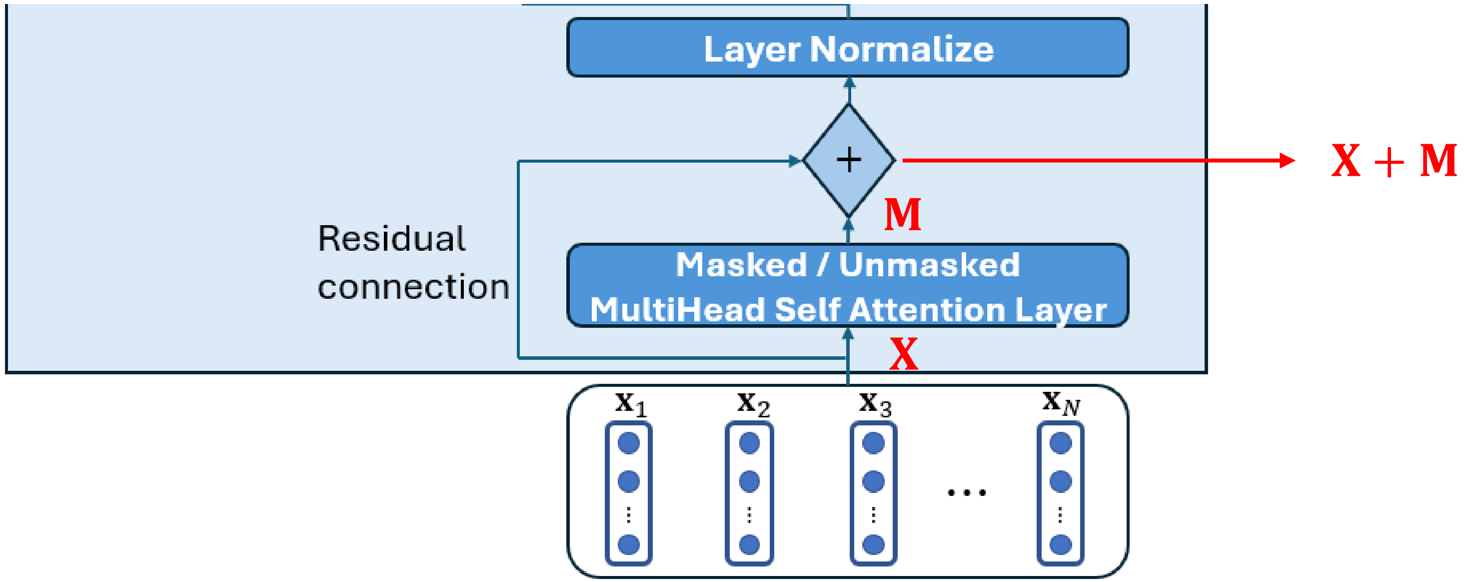
The true power of attention was first explored in the well known "Attention is all you need" paper released in 2017. The authors proposed a network architecture called the ***Transformer*** which was solely based on the attention mechanism. This architecture is now the basis for Large Language Models (LLMs)



Transformer

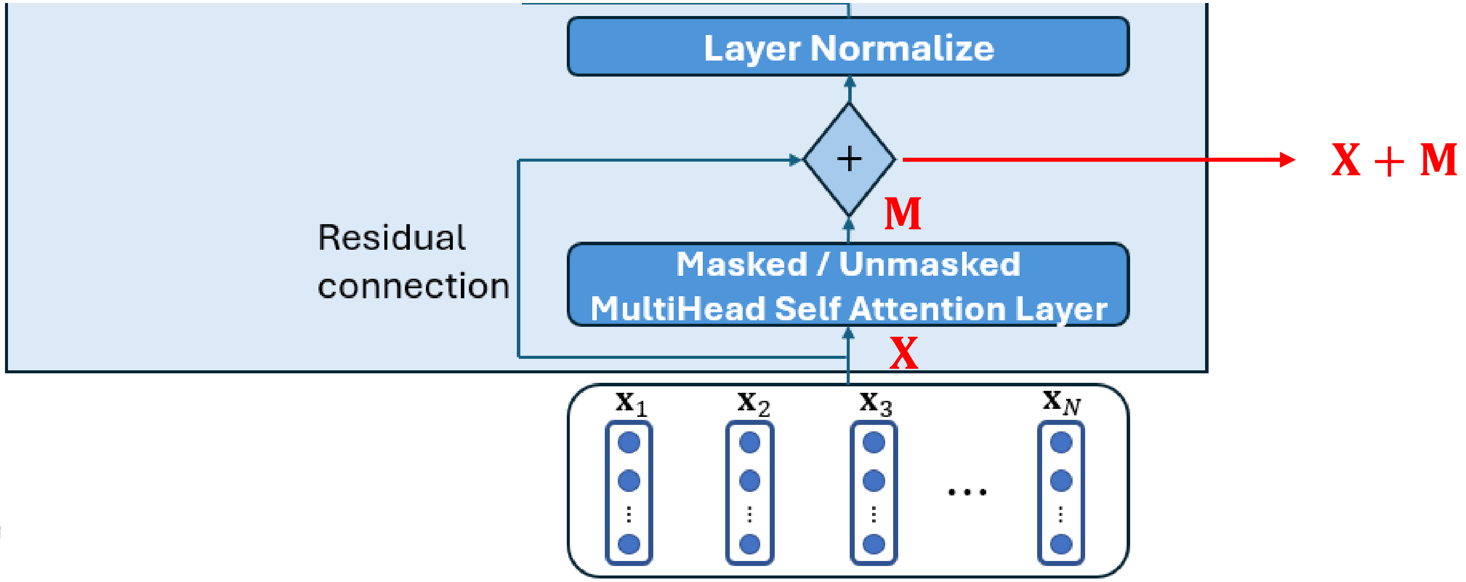
- Residual Connection
  - In deep networks, residual connections are connections that pass information from a lower layer to a higher layer without going through the intermediate layer.
  - Residual connections in transformers are implemented by adding a layer's input vector to its output vector before passing it forward.

$$\mathbf{X} + \text{output of MultiHead self attention}$$



Transformer

- Residual Connection
  - Residual connections in transformers are implemented by adding a layer's input vector to its output vector before passing it forward.



Input Embeddings

$$\mathbf{X} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Output of Attention

$$\mathbf{M} = \begin{bmatrix} 0.5 & 1.0 & 1.5 \\ 2.0 & 2.5 & 3.0 \\ 3.5 & 4.0 & 4.5 \end{bmatrix}$$

$$\mathbf{X}+\mathbf{M} = \mathbf{H} = \begin{bmatrix} 1.5 & 3.0 & 4.5 \\ 6.0 & 7.5 & 9.0 \\ 10.5 & 12.0 & 13.5 \end{bmatrix}$$

Transformer

- Layer Normlalization
  - Layer normalization (LayerNorm) is a normalization technique applied across the **features** of a vector. It standardizes the input by normalizing its mean and variance for each input vector independently.
  - The input to layer norm is a single vector of dimensionality d and the output is that vector normalized, again of dimensionality d

Transformer

- Layer Normlalization

  - Layer normalization (LayerNorm) is a normalization technique applied across the **features** of a vector. It standardizes the input by normalizing its mean and variance for each input vector independently.

  - The input to layer norm is a single vector $\mathbf{x}$ of dimensionality d and the output is that vector normalized, again of dimensionality d

$$\mu = \frac{1}{d}\sum_{i=1}^{d} x_i$$

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

$$\hat{\mathbf{X}} = \frac{(\mathbf{x} - \mu)}{\sigma}$$

$$\mathrm{LayerNorm}(\mathbf{x}) = \boldsymbol{\gamma}\hat{\mathbf{X}} + \boldsymbol{\beta}$$

Transformer

- Layer Normalization

Assume the sum of input embeddings and the output of the multihead attention layer is

$$\mathbf{X} + \mathbf{M} = \mathbf{H} = \begin{bmatrix} 1.5 & 3.0 & 4.5 \\ 6.0 & 7.5 & 9.0 \\ 10.5 & 12.0 & 13.5 \end{bmatrix}$$

$$\mathbf{h_1} = \begin{pmatrix} 1.5 \\ 3.0 \\ 4.5 \end{pmatrix}, \qquad \mathbf{h_2} = \begin{pmatrix} 6.0 \\ 7.5 \\ 9.0 \end{pmatrix}, \qquad \mathbf{h_3} = \begin{pmatrix} 10.5 \\ 12.0 \\ 13.5 \end{pmatrix}$$

$$\mu_1 = \frac{1.5 + 3.0 + 4.5}{3} = 3, \qquad \mu_2 = \frac{6.0 + 7.5 + 9.0}{3} = 7.5, \qquad \mu_3 = \frac{10.5 + 12.0 + 13.5}{3} = 12$$

$$\sigma_1 = \sqrt{\frac{(1.5-3)^2 + (3-3)^2 + (4.5-3)^2}{3}} = 1.22, \qquad \sigma_2 = \sqrt{\frac{(6-7.5)^2 + (7.5-7.5)^2 + (9-7.5)^2}{3}} = 1.22$$

$$\sigma_3 = \sqrt{\frac{(10.5-12)^2 + (12-12)^2 + (13.5-12)^2}{3}} = 1.22$$

Transformer

- Layer Normalization

Assume the sum of input embeddings and the output of the multihead attention layer is

$$\mathbf{h_1} = \begin{pmatrix} 1.5 \\ 3.0 \\ 4.5 \end{pmatrix}, \quad \mathbf{h_2} = \begin{pmatrix} 6.0 \\ 7.5 \\ 9.0 \end{pmatrix}, \quad \mathbf{h_3} = \begin{pmatrix} 10.5 \\ 12.0 \\ 13.5 \end{pmatrix}$$

$$\widehat{\mathbf{h_1}} = \begin{pmatrix} (1.5-3)/1.22 \\ (3-3)/1.22 \\ (4.5-3)/1.22 \end{pmatrix}, \quad \widehat{\mathbf{h_1}} = \begin{pmatrix} (6-7.5)/1.22 \\ (7.5-7.5)/1.22 \\ (9-7.5)/1.22 \end{pmatrix}, \quad \widehat{\mathbf{h_1}} = \begin{pmatrix} (10.5-12)/1.22 \\ (12-12)/1.22 \\ (13.5-12)/1.22 \end{pmatrix}$$

$$\widehat{\mathbf{h_1}} = \begin{pmatrix} -1.23 \\ 0 \\ 1.23 \end{pmatrix}, \quad \widehat{\mathbf{h_1}} = \begin{pmatrix} -1.23 \\ 0 \\ 1.23 \end{pmatrix}, \quad \widehat{\mathbf{h_1}} = \begin{pmatrix} -1.23 \\ 0 \\ 1.23 \end{pmatrix}$$

$$\text{layerNorm}(\mathbf{h_1}) = \boldsymbol{\gamma}\widehat{\boldsymbol{h_1}} + \boldsymbol{\beta}, \qquad \text{layerNorm}(\mathbf{h_2}) = \boldsymbol{\gamma}\widehat{\boldsymbol{h_2}} + \boldsymbol{\beta}, \qquad \text{layerNorm}(\mathbf{h_3}) = \boldsymbol{\gamma}\widehat{\boldsymbol{h_3}} + \boldsymbol{\beta}$$

Transformer

- Layer Normalization

Assume the sum of input embeddings and the output of the multihead attention layer is

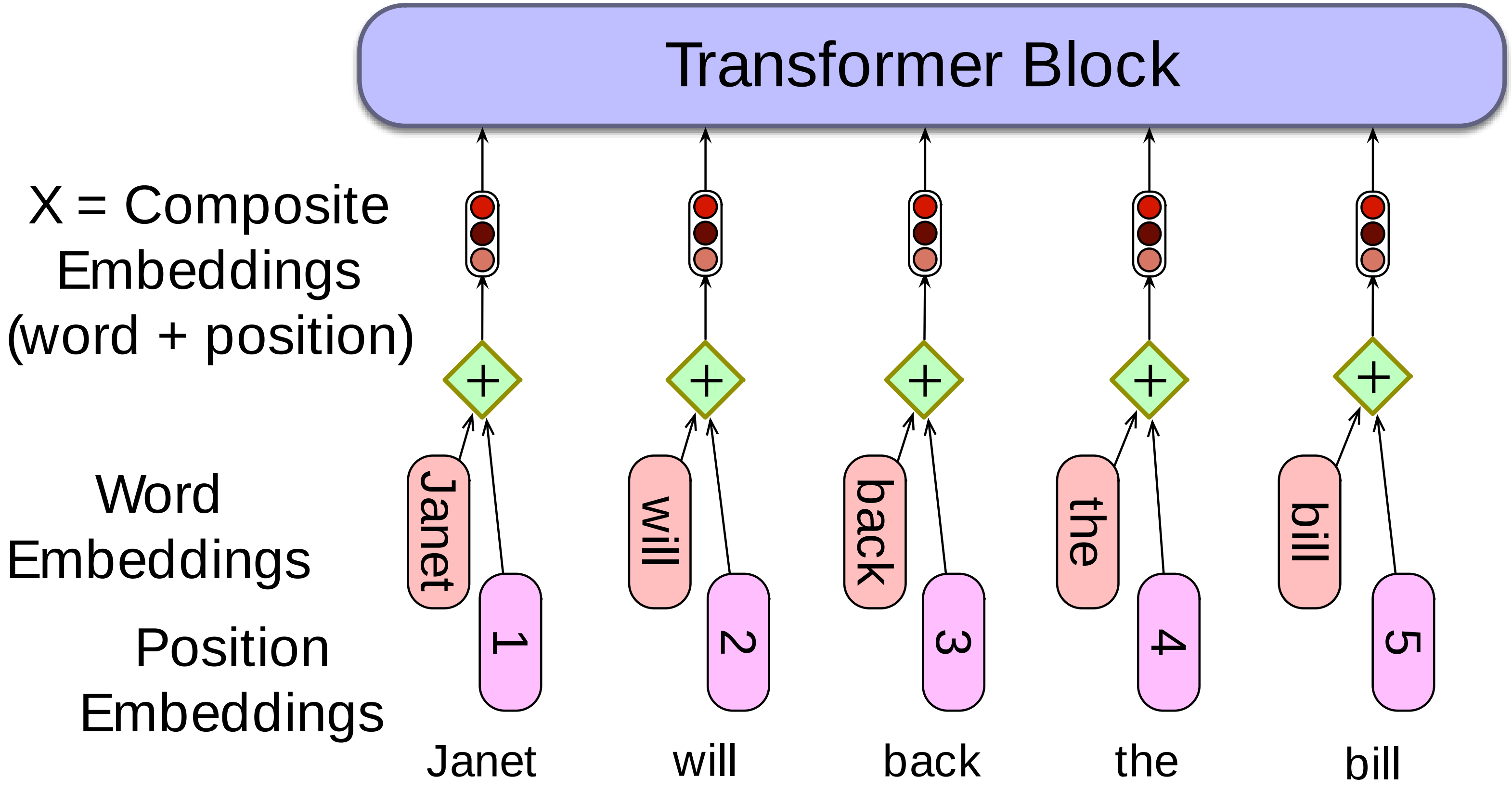
$$\widehat{\mathbf{h_1}} = \begin{pmatrix} -1.23 \\ 0 \\ 1.23 \end{pmatrix}, \qquad \widehat{\mathbf{h_1}} = \begin{pmatrix} -1.23 \\ 0 \\ 1.23 \end{pmatrix}, \qquad \widehat{\mathbf{h_1}} = \begin{pmatrix} -1.23 \\ 0 \\ 1.23 \end{pmatrix}$$

For simplicity: $\boldsymbol{\gamma} = \mathbf{1}$ and $\boldsymbol{\beta} = \mathbf{0}$

$$\text{layerNorm}(\mathbf{h_1}) = \widehat{\boldsymbol{h_1}}, \qquad \text{layerNorm}(\mathbf{h_2}) = \widehat{\boldsymbol{h_2}}, \qquad \text{layerNorm}(\mathbf{h_3}) = \widehat{\boldsymbol{h_3}}$$

Positional Embeddings

- The matrix $\mathbf{X}$ (of shape $[N \times d]$) has an embedding for each word in the context.
- This embedding is created by adding two distinct embedding for each input
  - token embedding
  - positional embedding



Language Modeling using Transformer