CSCE 638 Natural Language Processing Foundation and Techniques

Lecture 8: Transformers

Kuan-Hao Huang

Spring 2026

(Some slides adapted from Chris Manning, Karthik Narasimhan, Danqi Chen, and Vivian Chen)

Course Project

- Each team should have 3–4 members of your choice
- Two possible tracks of projects
  - Research Track
  - Application Track
    - Application track will present first at the end of semester

Course Project – Research Track

- Example topics
  - Choose a topic by selecting an existing problem discussed in class and developing new ideas around it
  - Identify any unresolved challenges from a published paper and improve the proposed approach
  - Participate in ongoing shared tasks at SemEval, CoNLL, Kaggle, or relevant workshops, and present and discuss the techniques you apply
- If you are not sure whether your proposed topic is appropriate, please come to office hours to discuss it

Course Project – Application Track

- Example topics
  - Design a system with a user interface that applies NLP techniques to solve a real-world problem
  - Build a Chrome extension that applies NLP techniques to assist users in real time
  - Develop an App with compelling features that requires NLP techniques
- If you are not sure whether your proposed topic is appropriate, please come to office hours to discuss it

Course Project – Proposal

- Due: Mar 6
- Page limit: 2 pages (excluding reference)
- Format: ACL style
- More detailed requirements will be released closer to the due date

Computational Resources

- Texas A&M High Performance Research Computing (HPRC)
  - https://hprc.tamu.edu/resources/

System Name:	FASTER
Host Name:	faster.hprc.tamu.edu
Operating System:	Rocky Linux 8
Total Compute Cores/Nodes:	11,520 cores 180 nodes
Compute Nodes:	180 64-core compute nodes, each with 256GB RAM
Composable GPUs:	200 T4 16GB GPUs 40 A100 40GB GPUs 8 A10 24GB GPUs 4 A30 24GB GPUs 8 A40 48GB GPUs

System Name:	Grace
Host Name:	grace.hprc.tamu.edu
Operating System:	Linux (CentOS 7)
Total Compute Cores/Nodes:	45,376 cores 940 nodes
Compute Nodes:	800 48-core compute nodes, each with 384GB RAM 100 48-core GPU nodes, each with two A100 40GB GPUs and 384GB RAM 9 48-core GPU nodes, each with two RTX 6000 24GB GPUs and 384GB RAM 8 48-core GPU nodes, each with 4 T4 16GB GPUs 15 48-core GPU nodes, each with two A40 48GB GPUs and 384GB RAM 8 80-core large memory nodes, each with 3TB RAM

Team Sign-Up

- https://docs.google.com/spreadsheets/d/1qUZPFI4wciToJsXye8-WN4L7xVG38IWdS2GCCzmu84A/edit?usp=sharing

Team Match

Track	Broad Topic (feel free to add new ones!)	Related Readings
Research	LLM Hallucination	https://arxiv.org/abs/2305.13534 https://arxiv.org/abs/2303.08896 https://arxiv.org/abs/2407.07071
Research	In-Context Learning with LLMs	https://arxiv.org/abs/2202.12837 https://arxiv.org/abs/2402.05403 https://arxiv.org/abs/2311.00237
Research	Cross-Lingual Transfer Learning	https://arxiv.org/abs/1901.07291 https://arxiv.org/abs/1911.02116 https://arxiv.org/abs/1912.07840
Research	LLM Reasoning	https://arxiv.org/abs/2201.11903 https://arxiv.org/abs/2305.10601 https://arxiv.org/abs/2501.19393
Research	Math and Logical Reasoning	https://arxiv.org/abs/2410.05229 https://arxiv.org/abs/2405.18357 https://arxiv.org/abs/2410.15580
Research	Model Self-Correction	https://arxiv.org/abs/2203.11171 https://arxiv.org/abs/2303.17651 https://arxiv.org/abs/2211.00053 https://arxiv.org/abs/2310.01798
Research	Model Editing	https://arxiv.org/abs/2202.05262 https://arxiv.org/abs/2310.20138 https://arxiv.org/abs/2308.08742
Research	Adversarial Attacks for NLP Models	https://arxiv.org/abs/1804.07998 https://arxiv.org/abs/2004.09984 https://arxiv.org/abs/2305.14710

	The following are proposed by students.						
Track	Topic	Related Readings	Interested! (Name)	Interested! (E-mail)	Interested! (Name)	Interested! (E-mail)	I

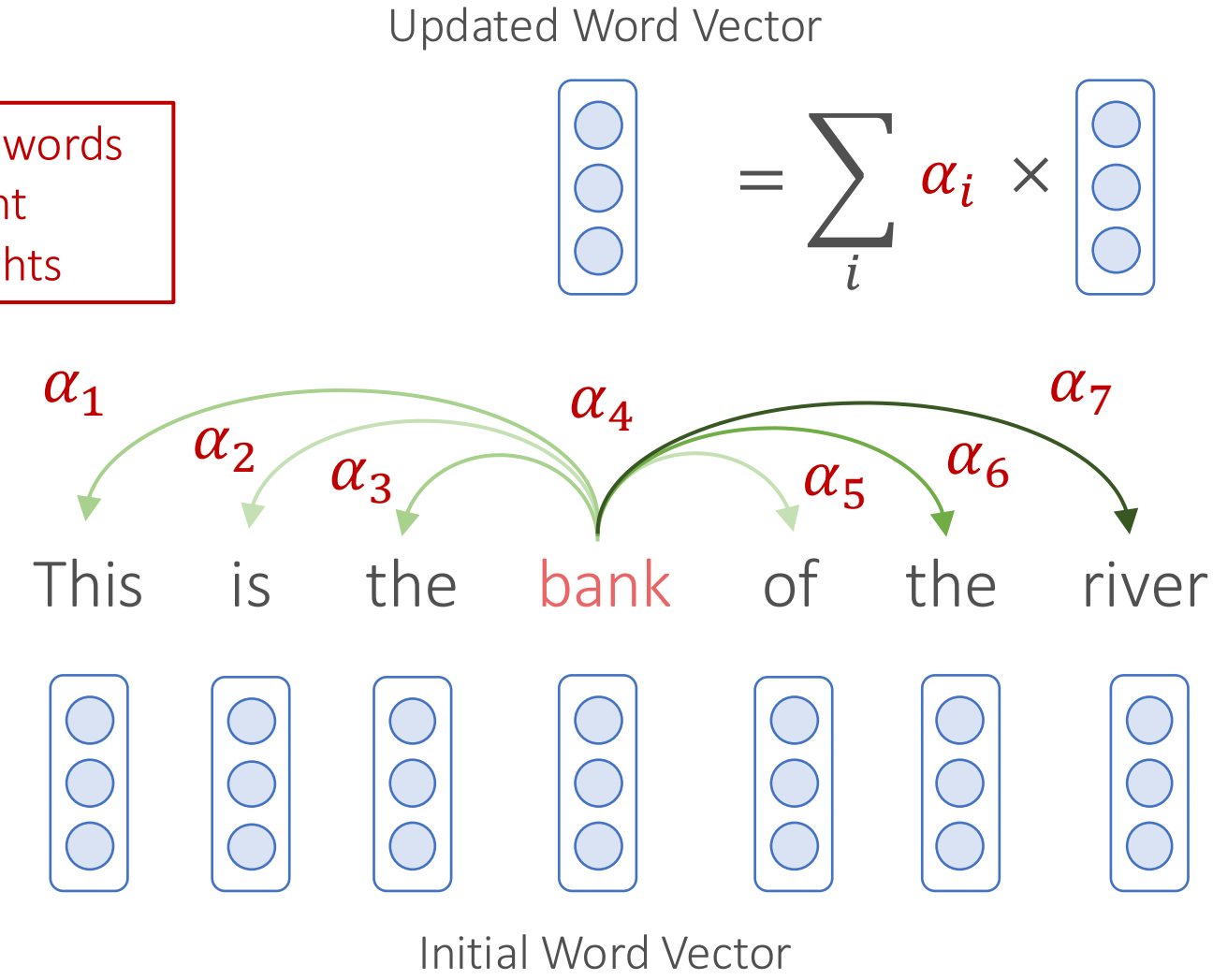
Questions?

Lecture Plan

- Transformers
  - Encoder
  - Decoder
  - Encoder-Decoder
- Transformers Variants
  - Longformer
  - Relative Positional Encoding
  - RoFormer

Recap: Transformer

Different context words have different **attention** weights



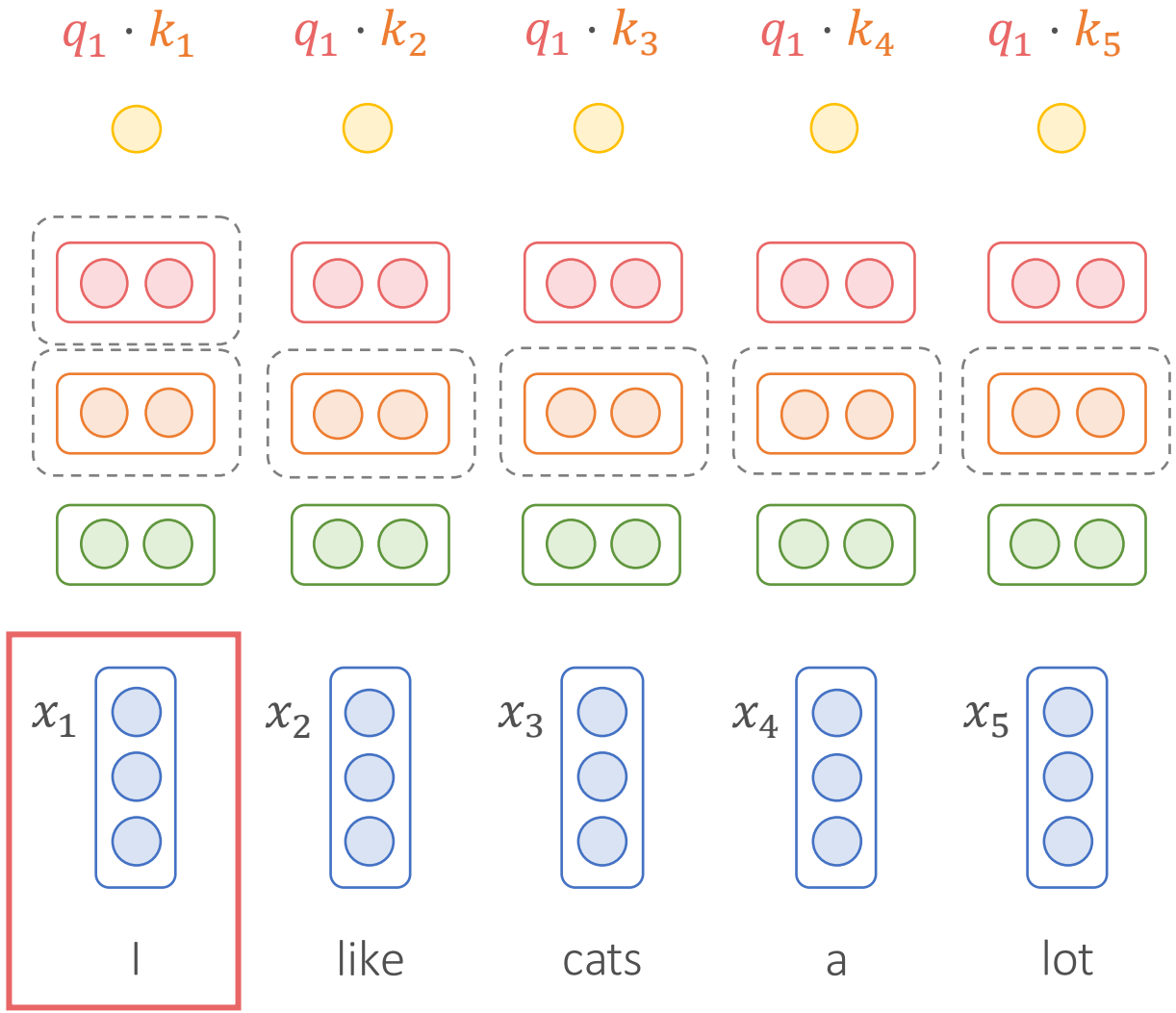
Recap: Transformer

Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Recap: Transformer

$$\alpha_{1,i} = \text{softmax}\left(\frac{q_1 \cdot k_i}{\sqrt{d}}\right)$$

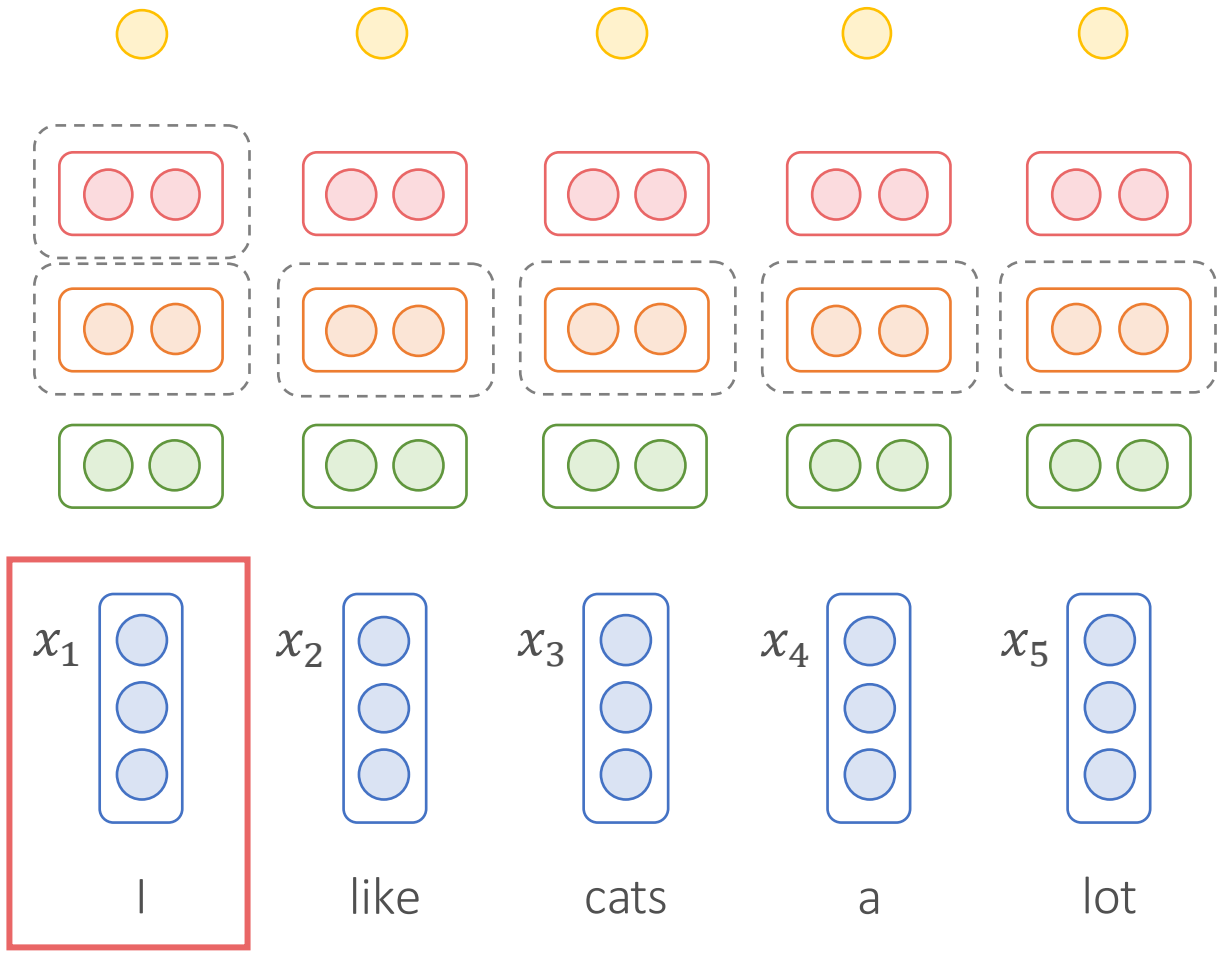
Vector dimension

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Recap: Transformer

Weighted Sum

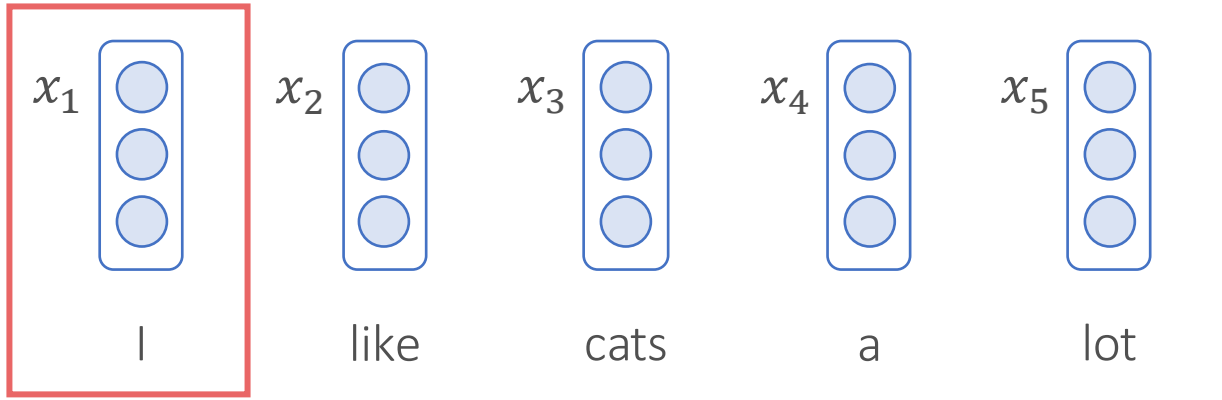
$$z_1 = \sum_i \alpha_{1,i} v_i$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Recap: Word Order?

Weighted Sum

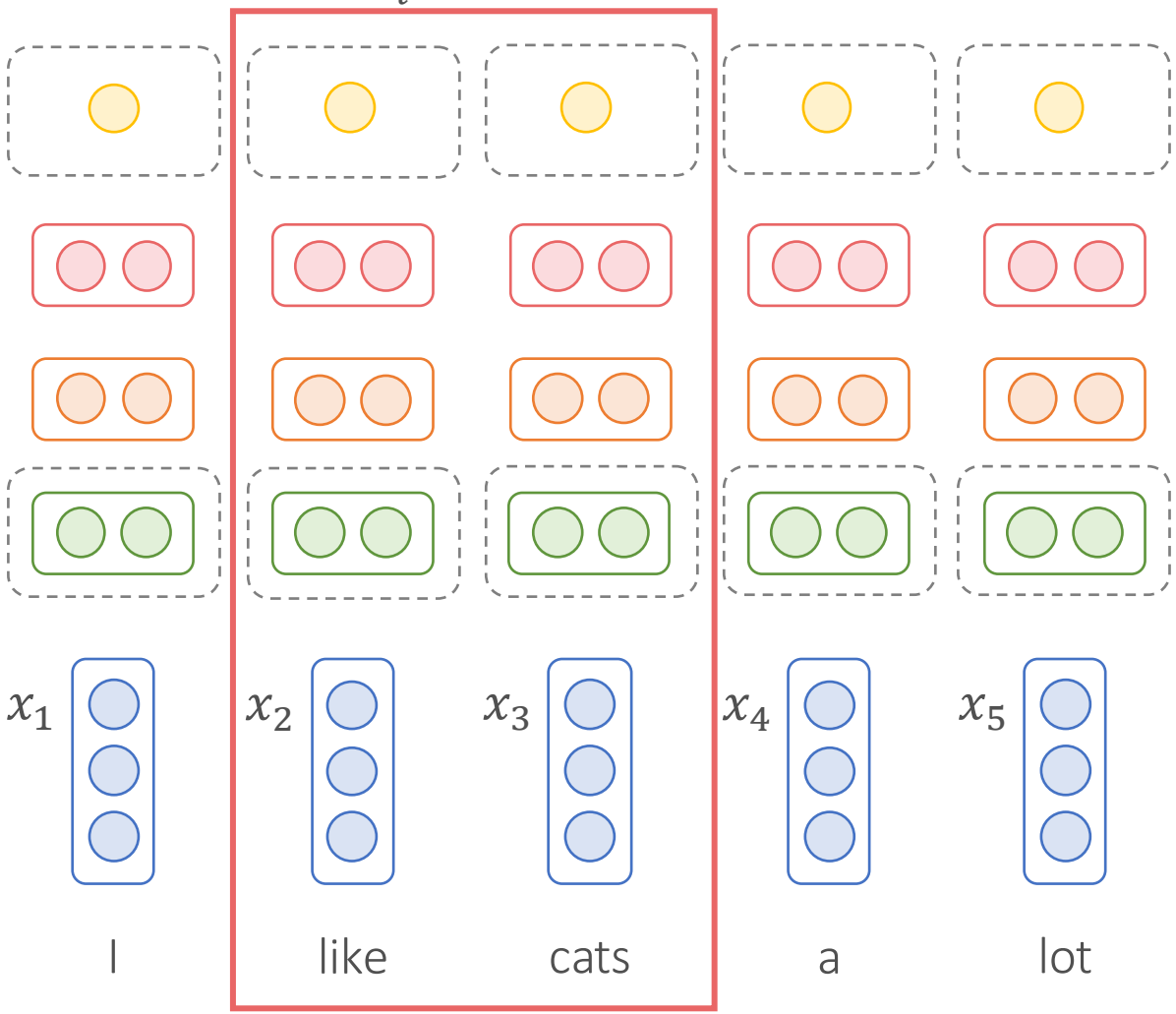
$$z_1 = \sum_i \alpha_{1,i} v_i$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Recap: Word Order?

Weighted Sum

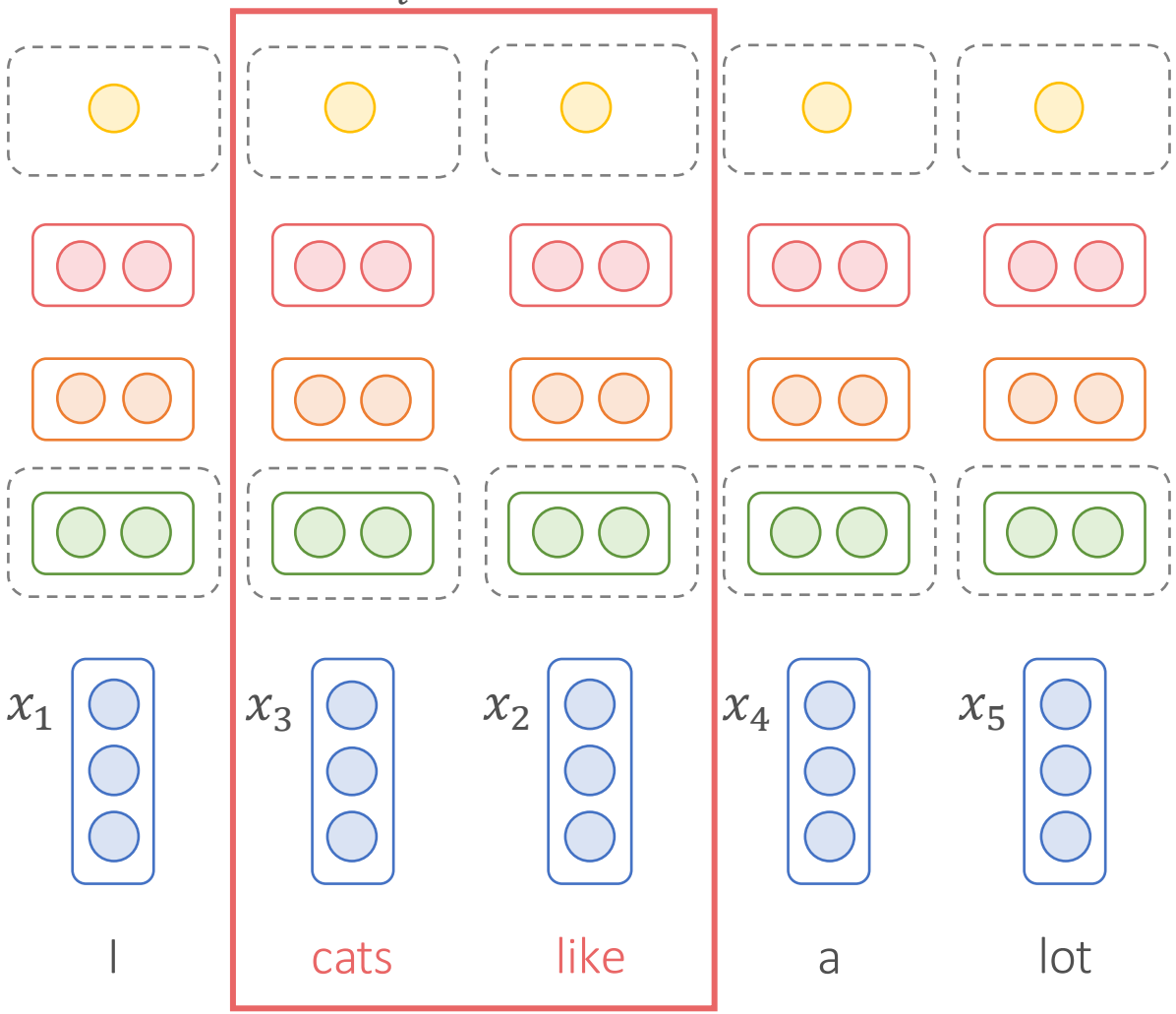
$$z_1 = \sum_i \alpha_{1,i} v_i$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

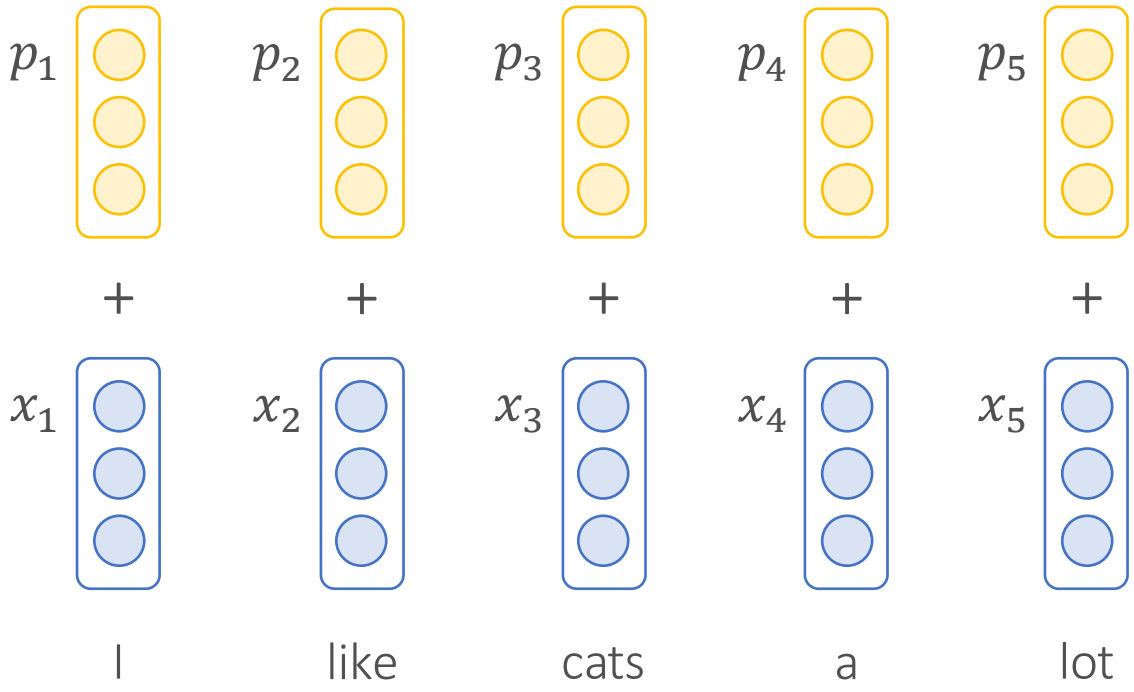
Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Recap: Positional Encoding

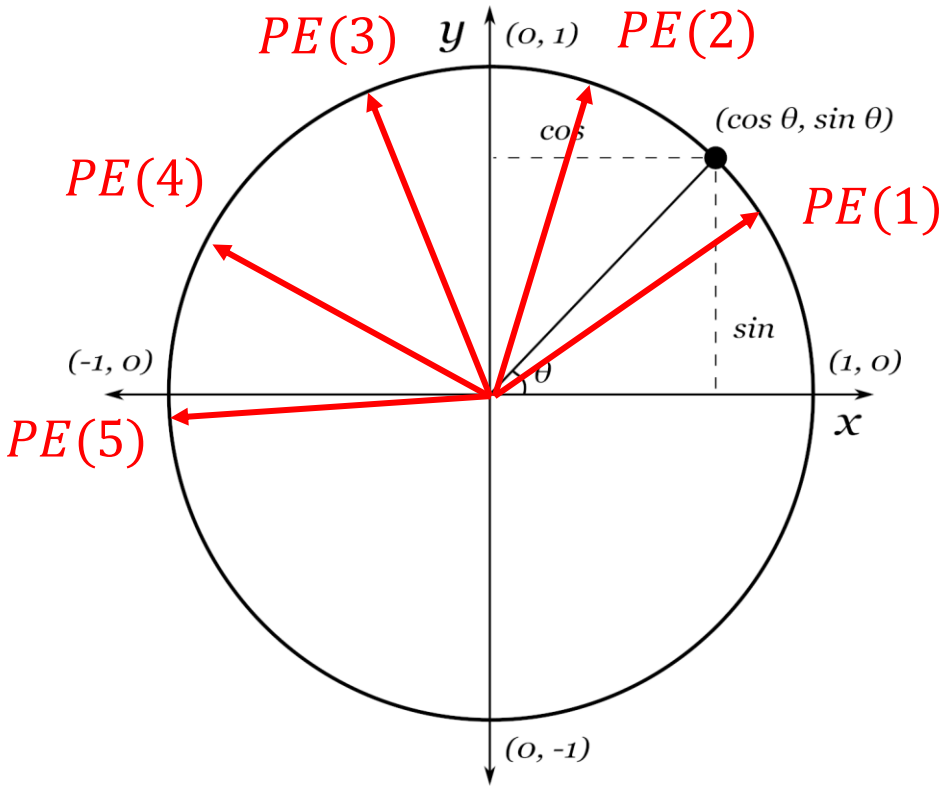
$$x_i \leftarrow x_i + PE_i$$



Recap: Sinusoidal Positional Encoding

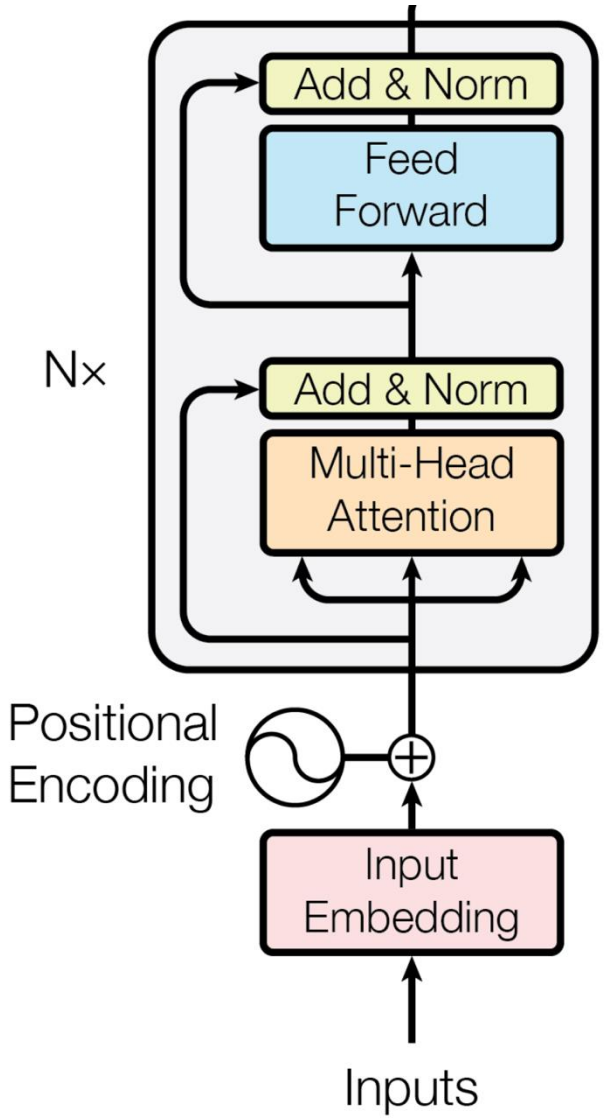
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



https://www.inchcalculator.com/unit-circle-calculator/

Recap: Transformer Encoder with Positional Encoding

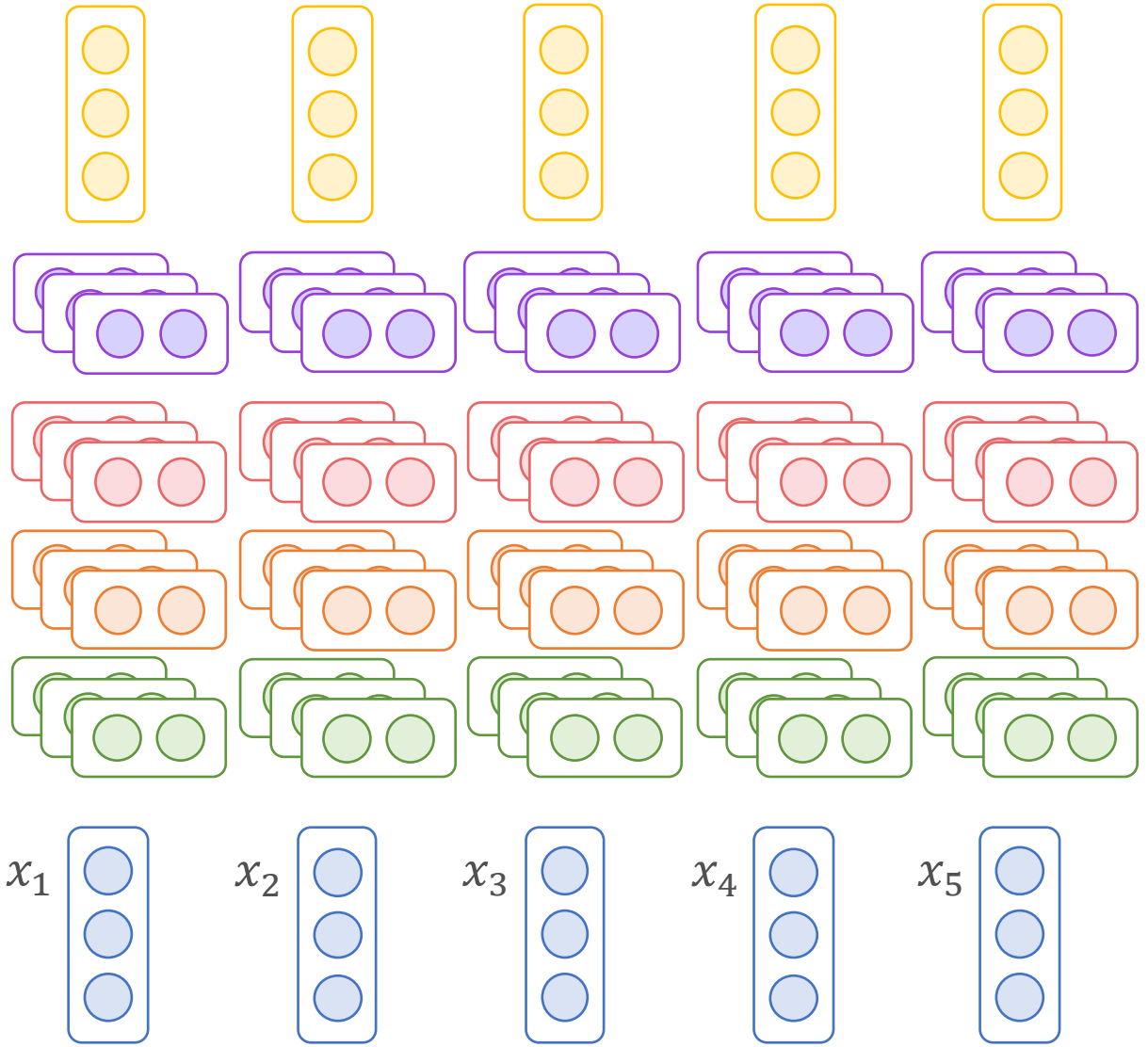


Transformer as Token-Level Encoder

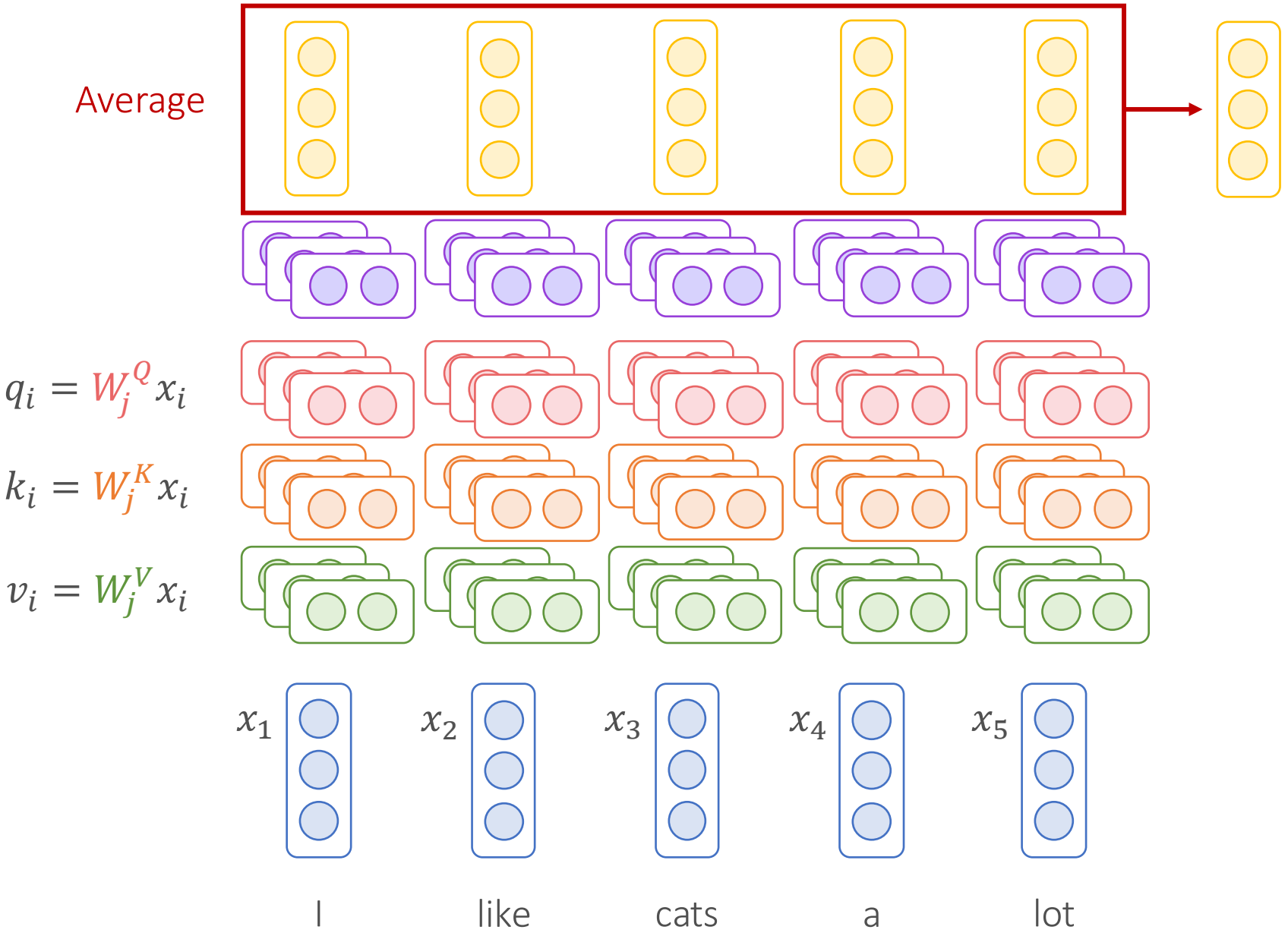
$q_i = W_j^Q x_i$

$k_i = W_j^K x_i$

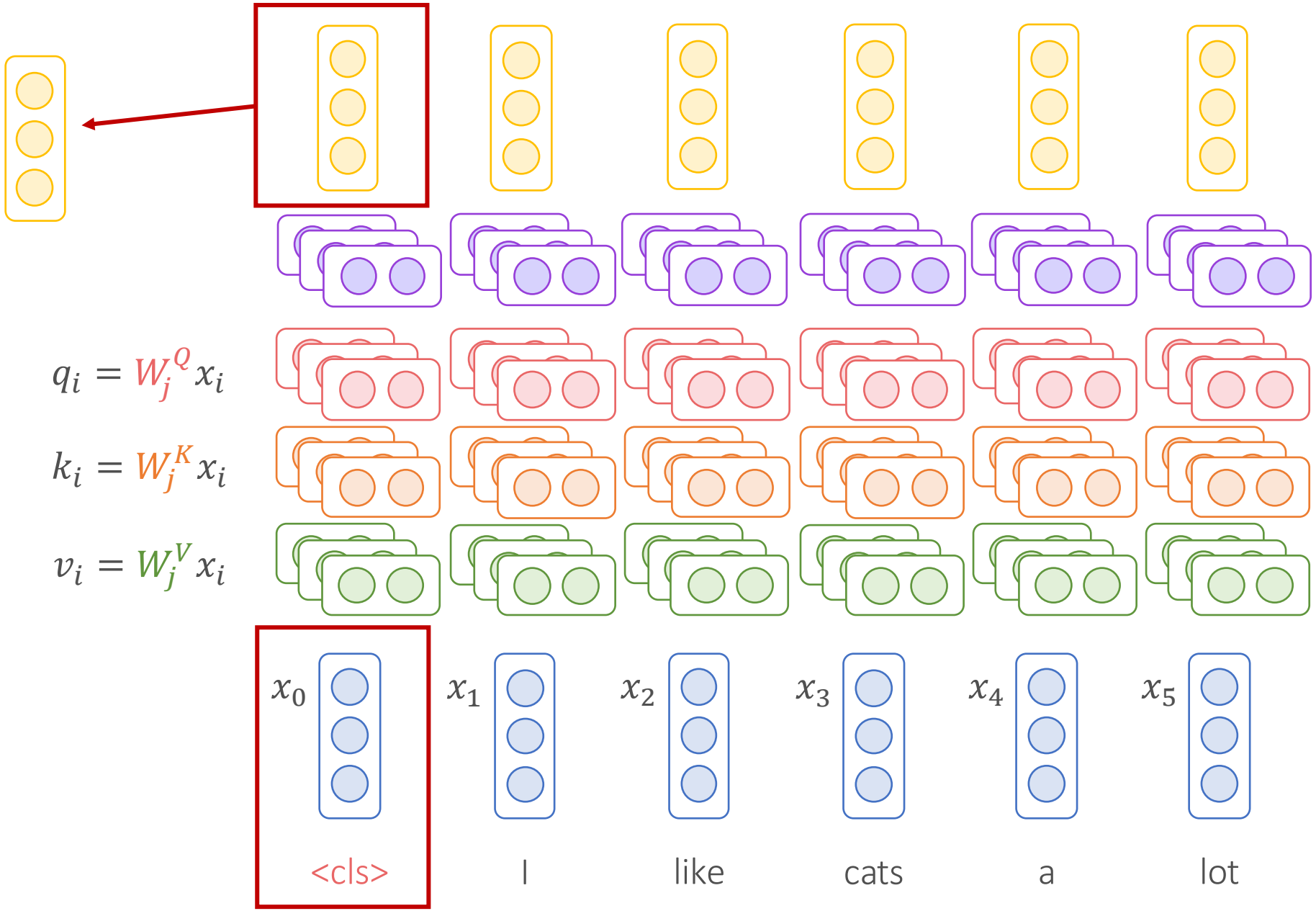
$v_i = W_j^V x_i$



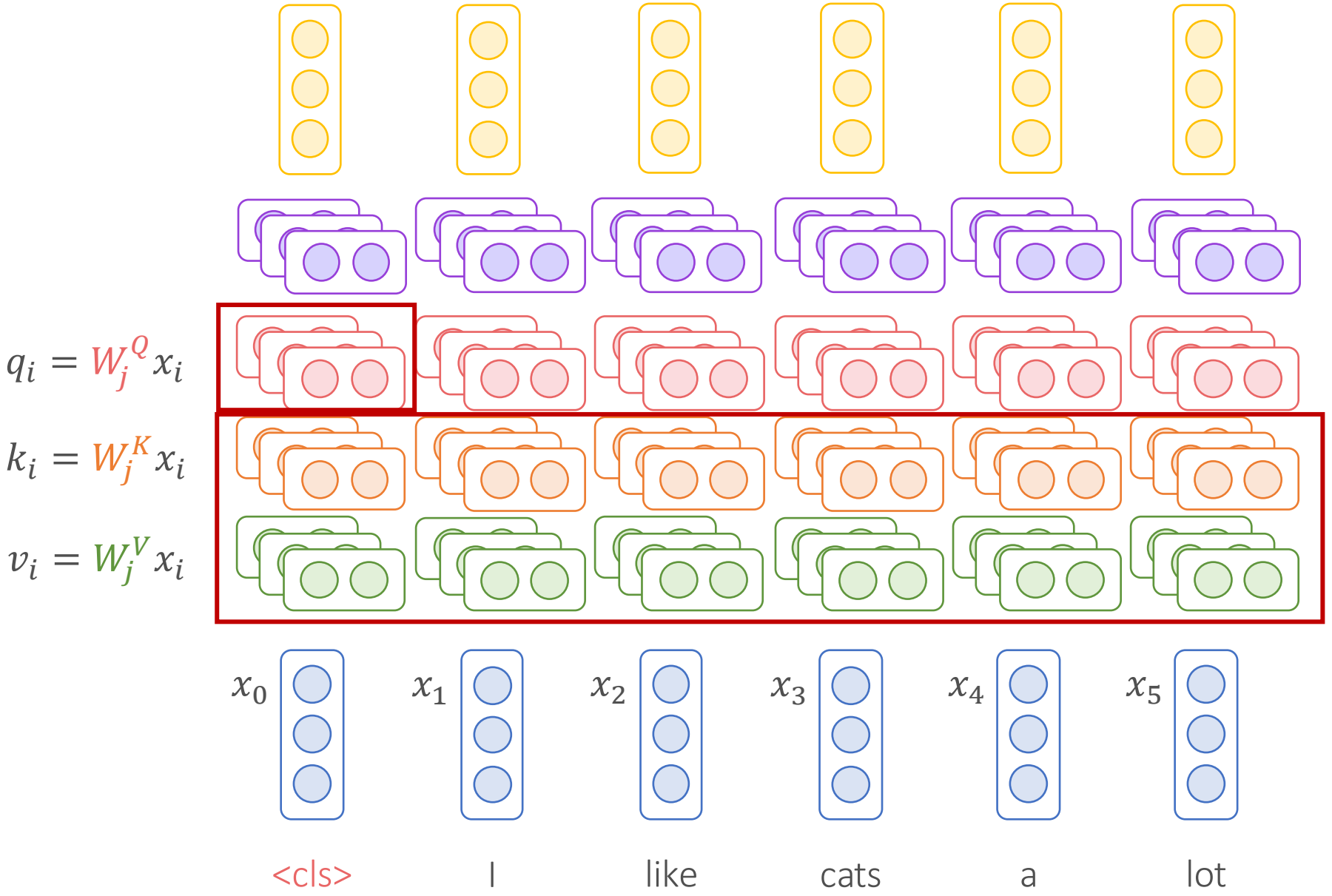
Transformer as Sentence-Level Encoder



Transformer as Sentence-Level Encoder



Transformer as Sentence-Level Encoder

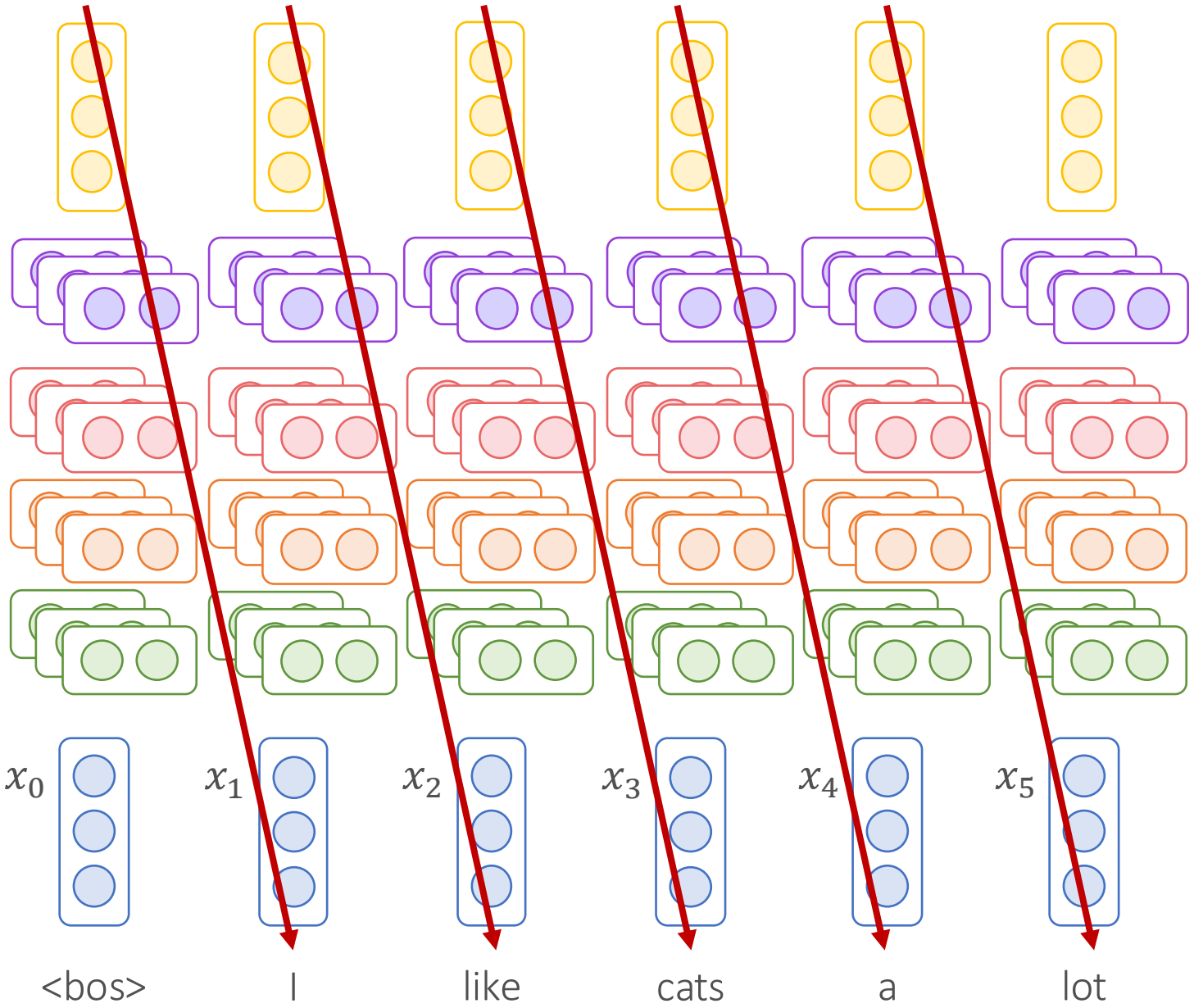


Transformer as Decoder?

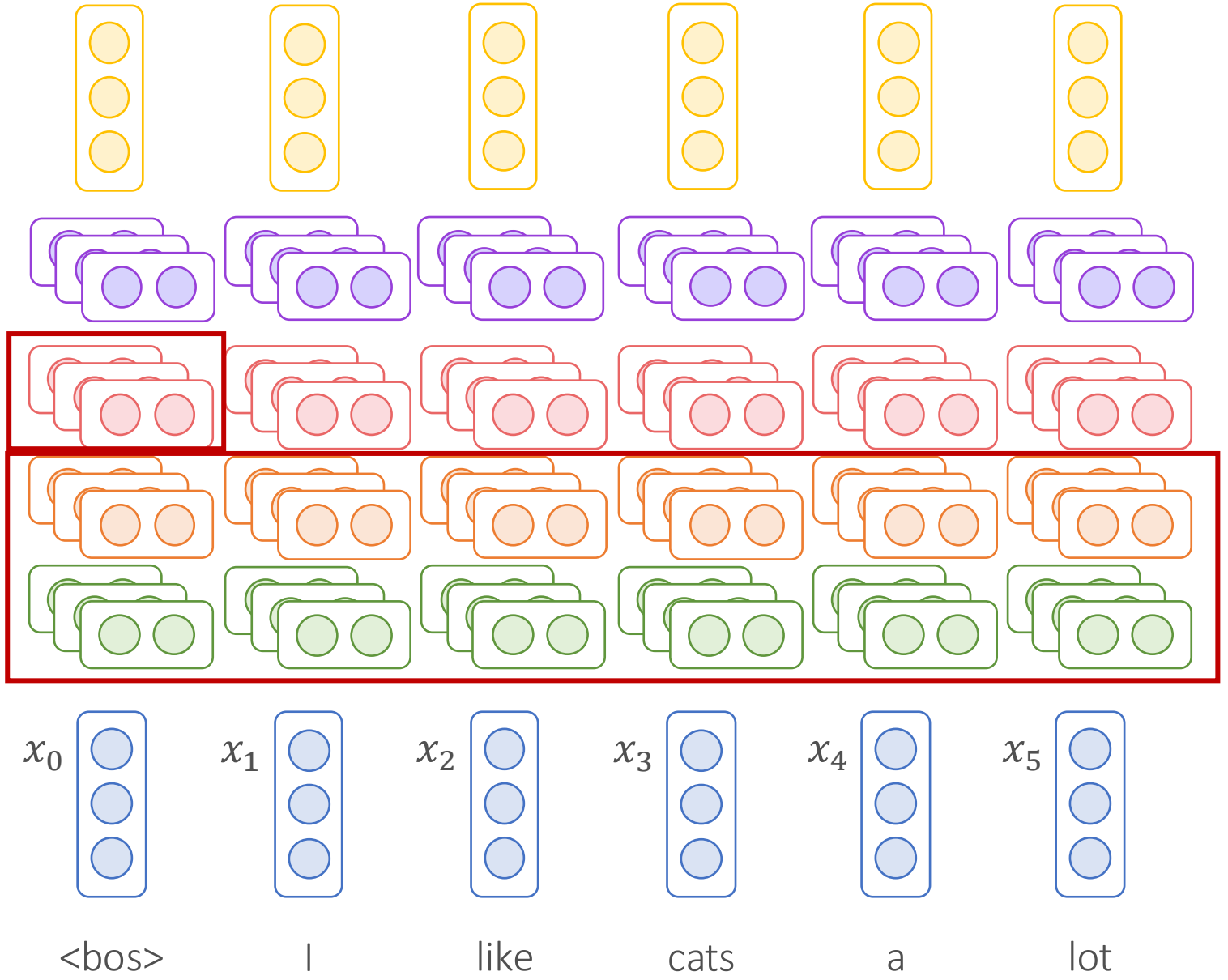
$q_i = W_j^Q x_i$

$k_i = W_j^K x_i$

$v_i = W_j^V x_i$

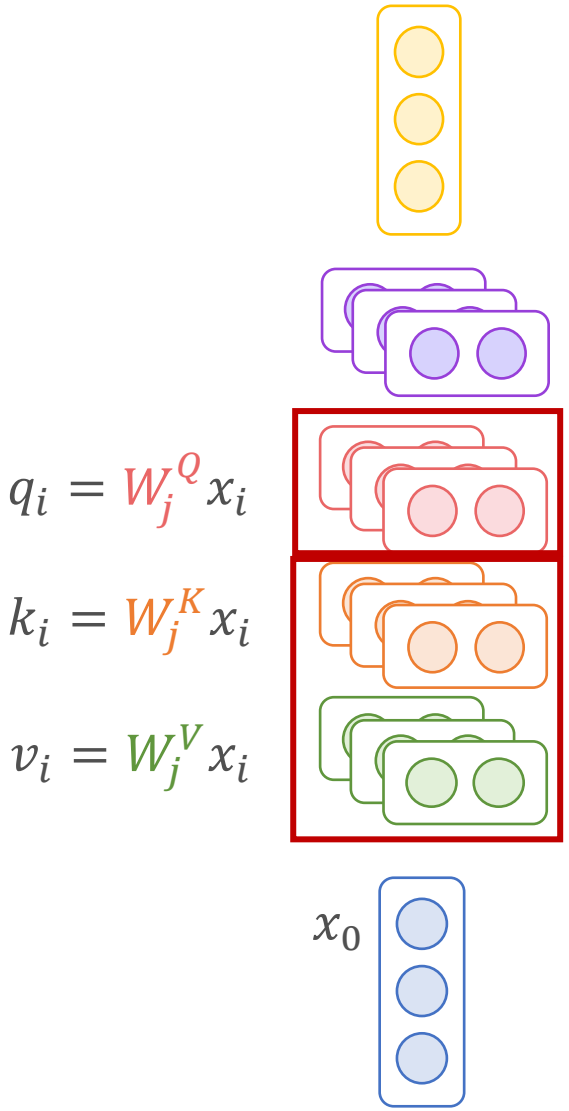


Transformer as Decoder?

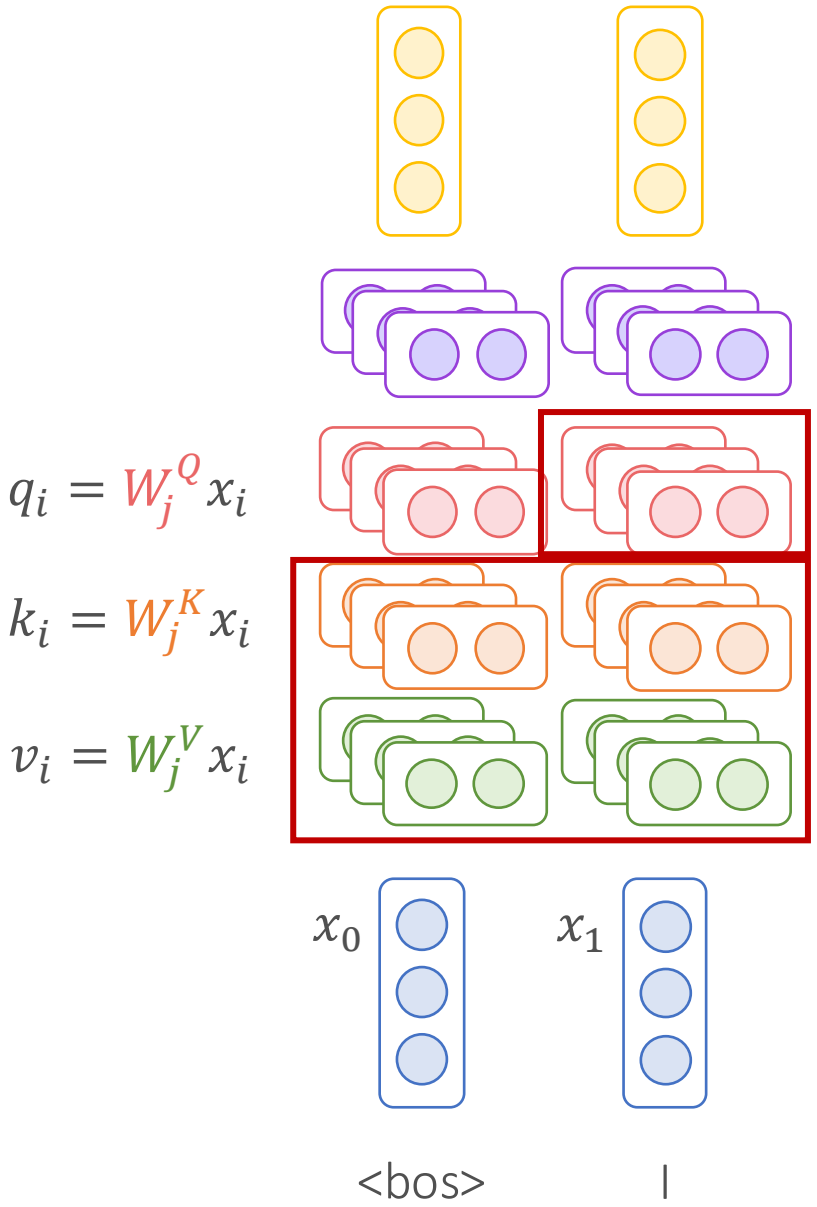


How to compute attention from the token we are going to generate?

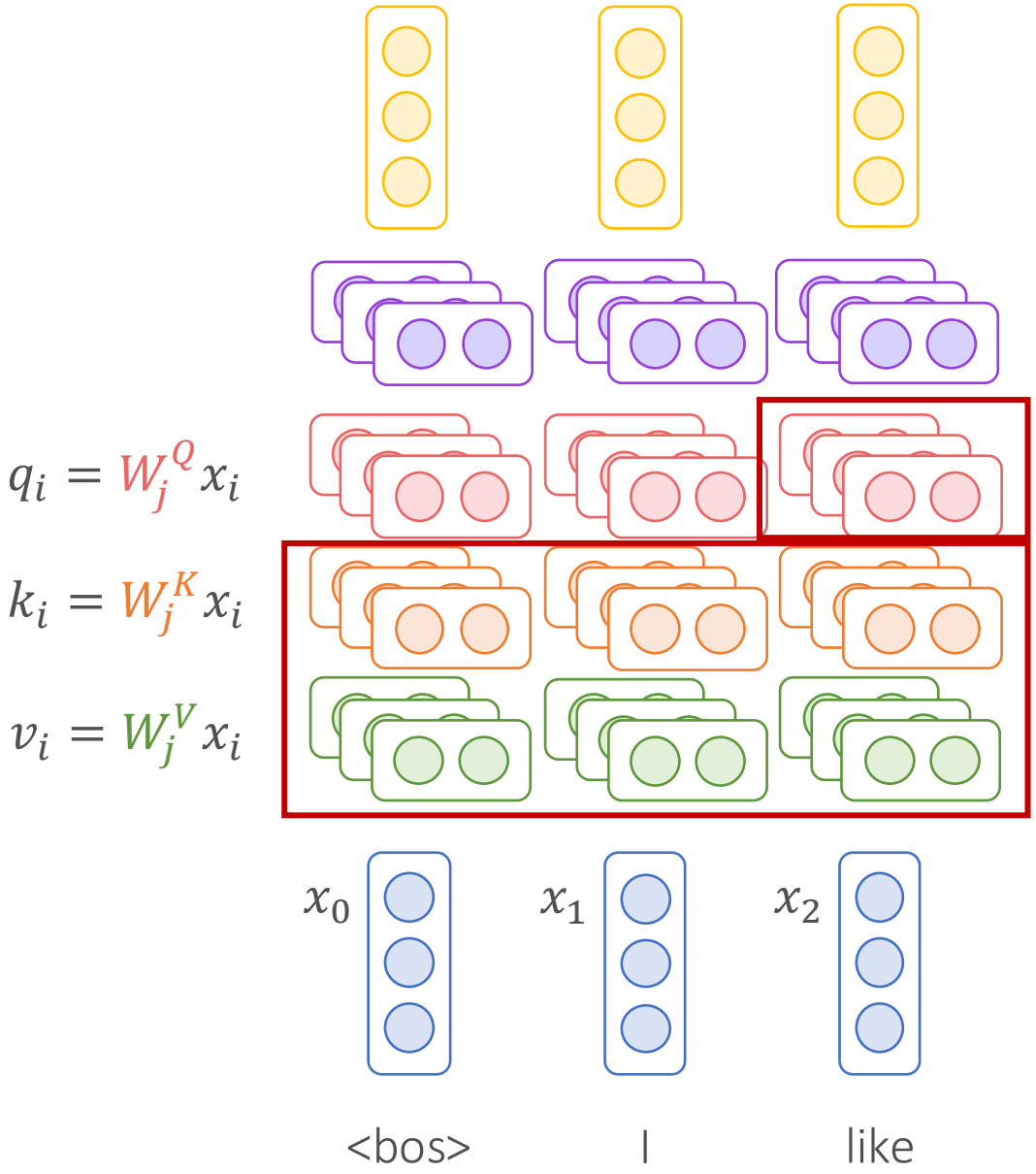
Transformer Decoder



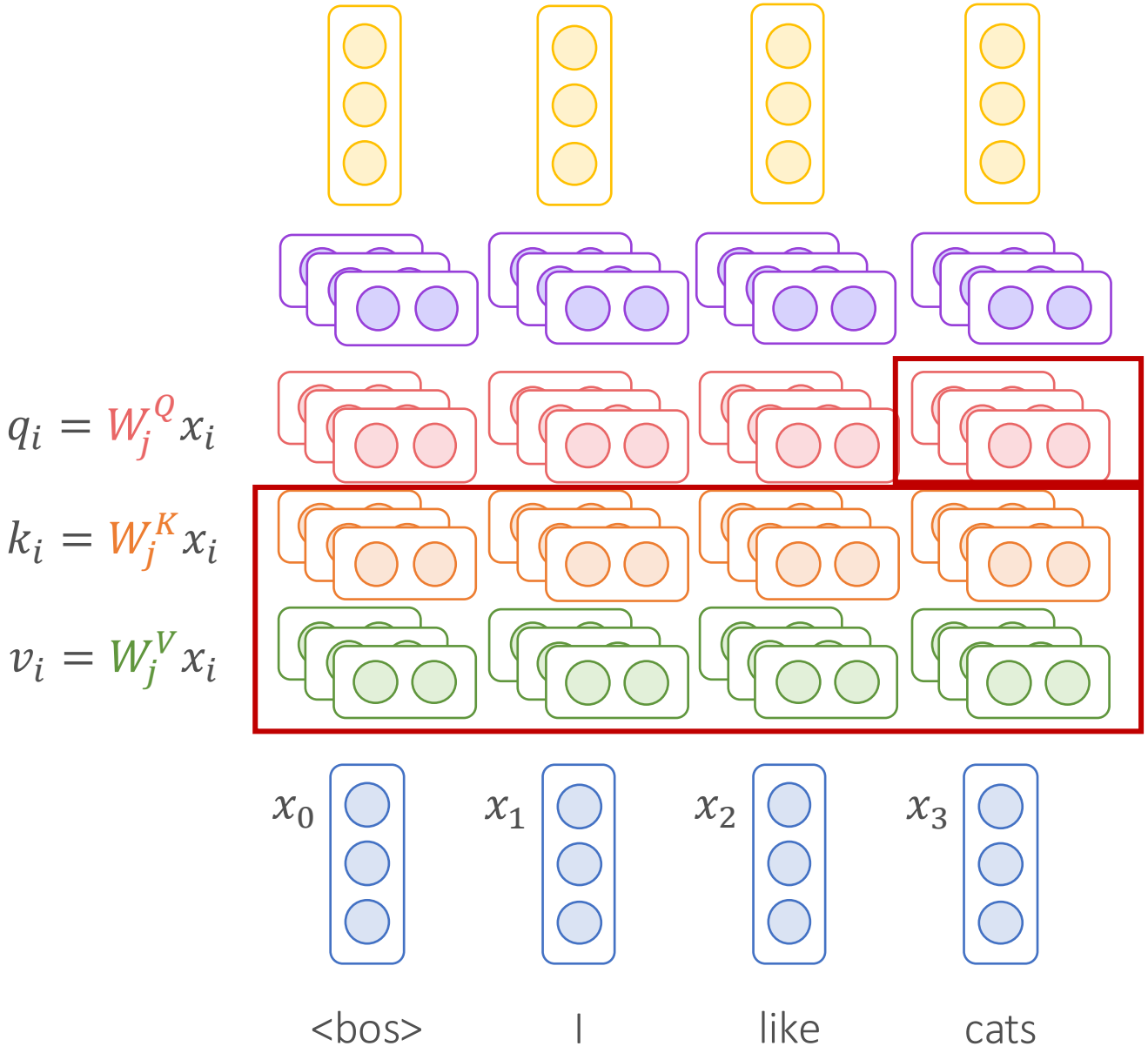
Transformer Decoder



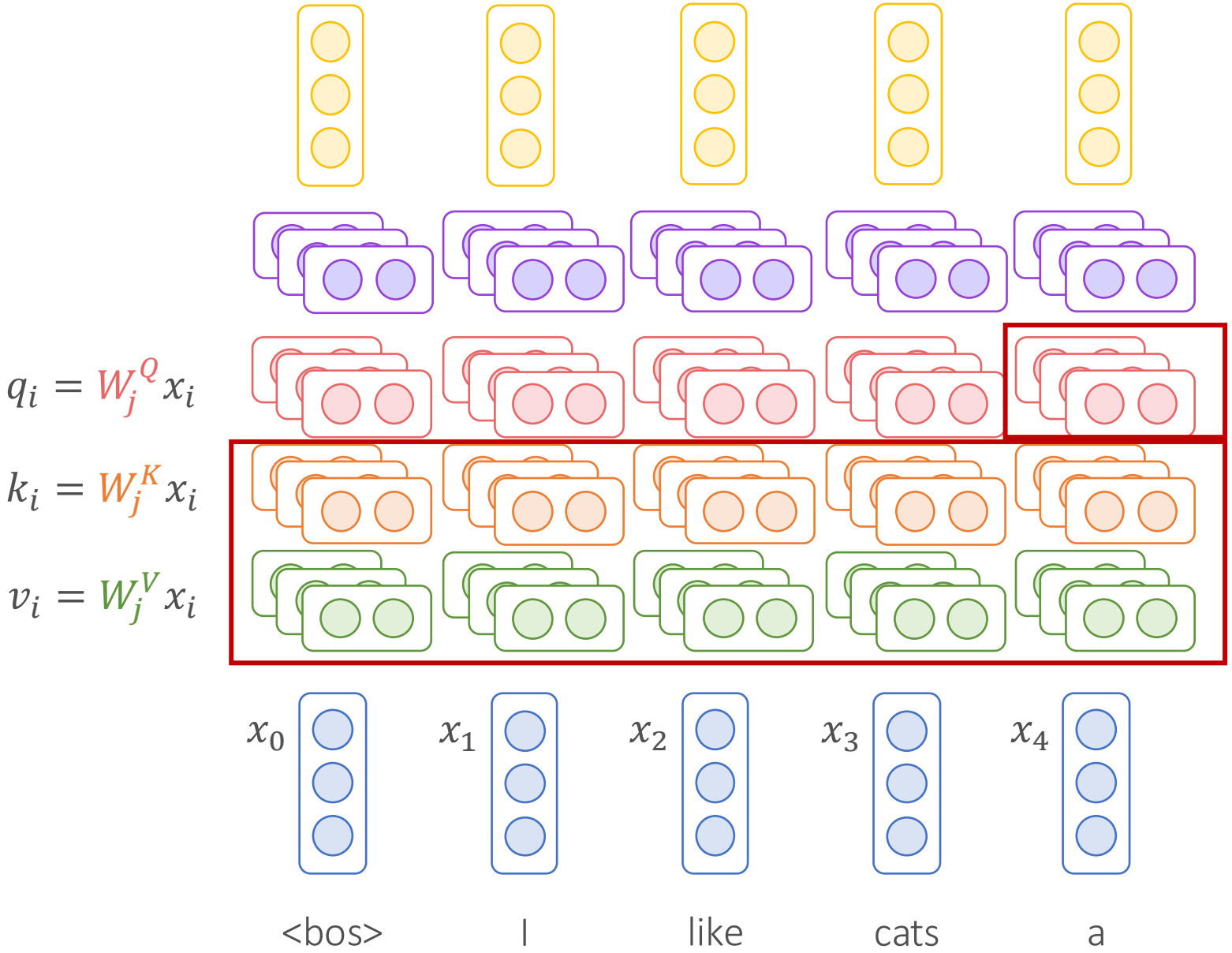
Transformer Decoder



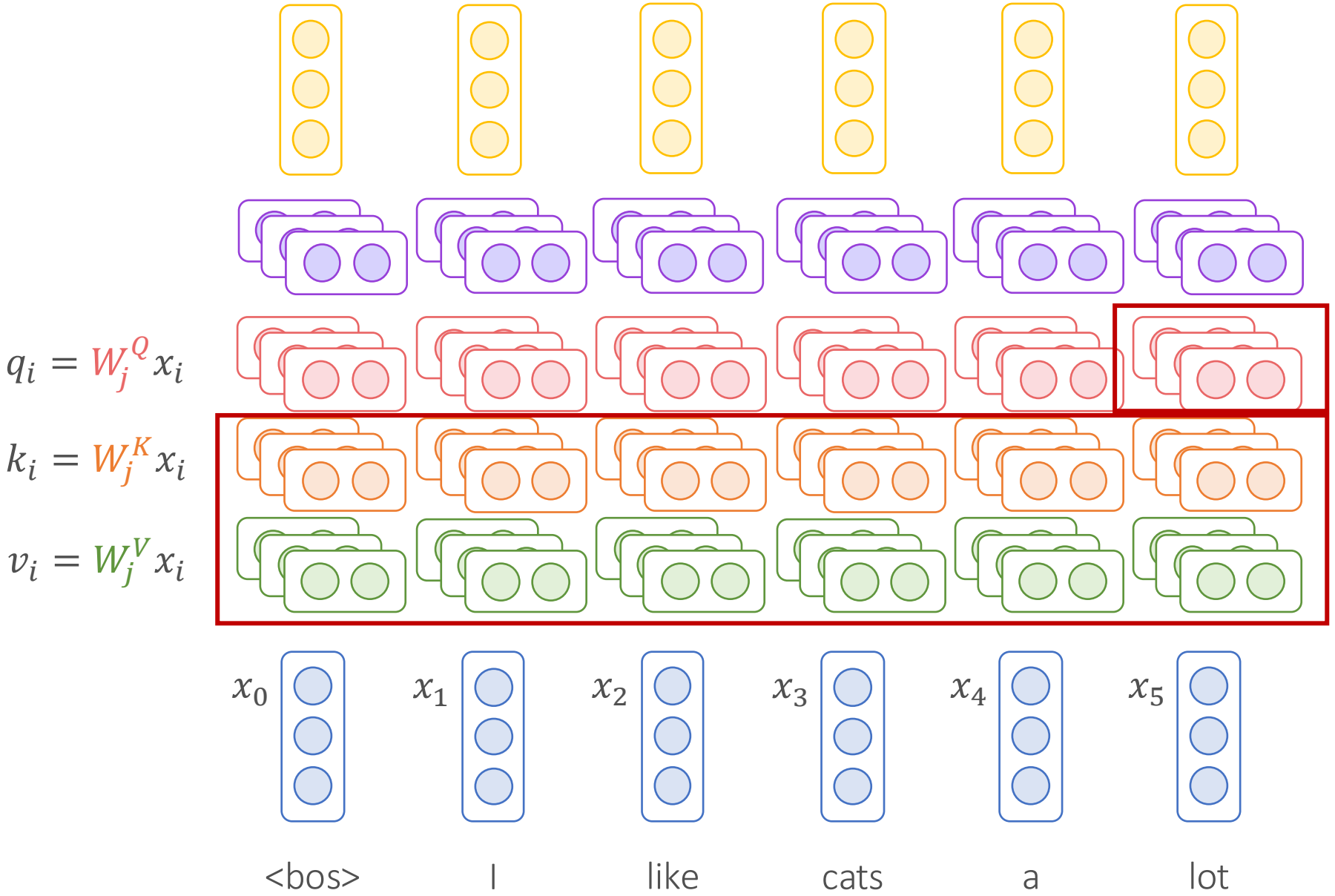
Transformer Decoder



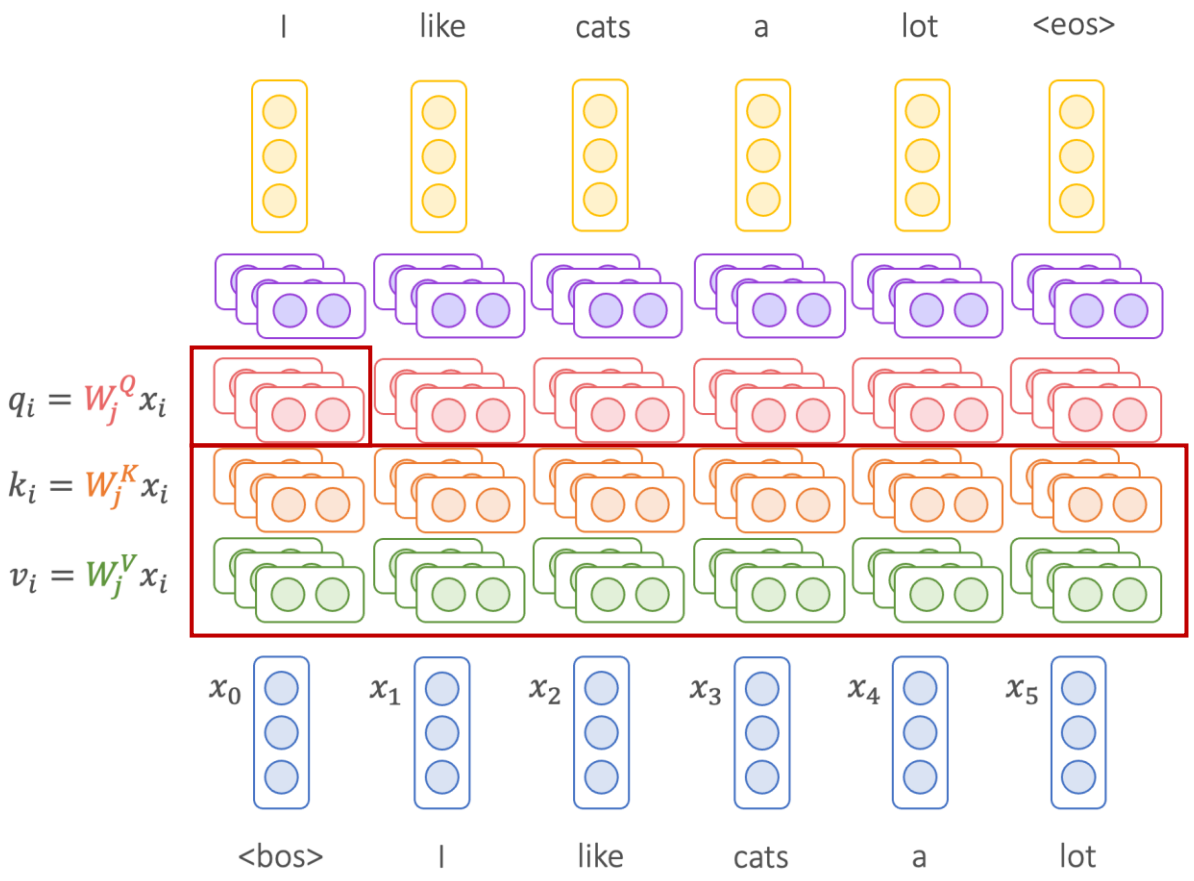
Transformer Decoder



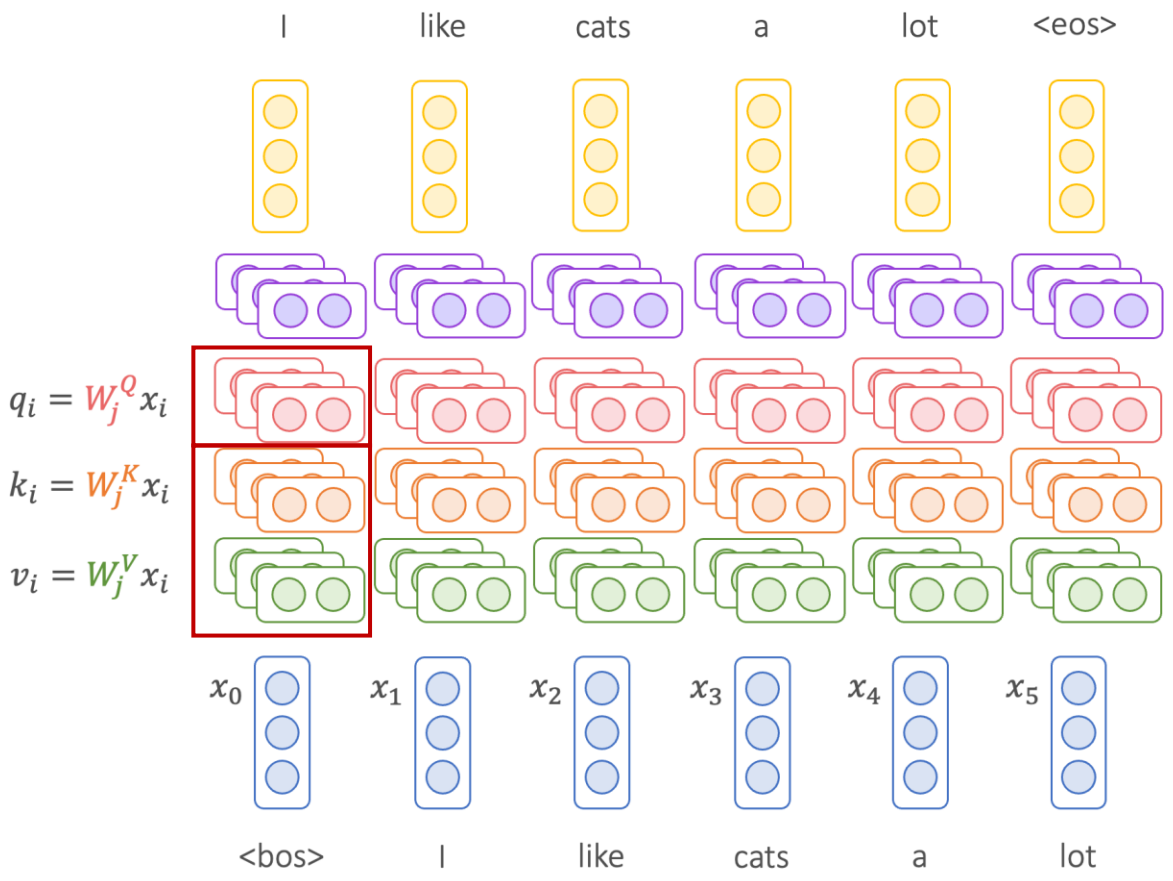
Transformer Decoder



Transformer Encoder vs. Transformer Decoder

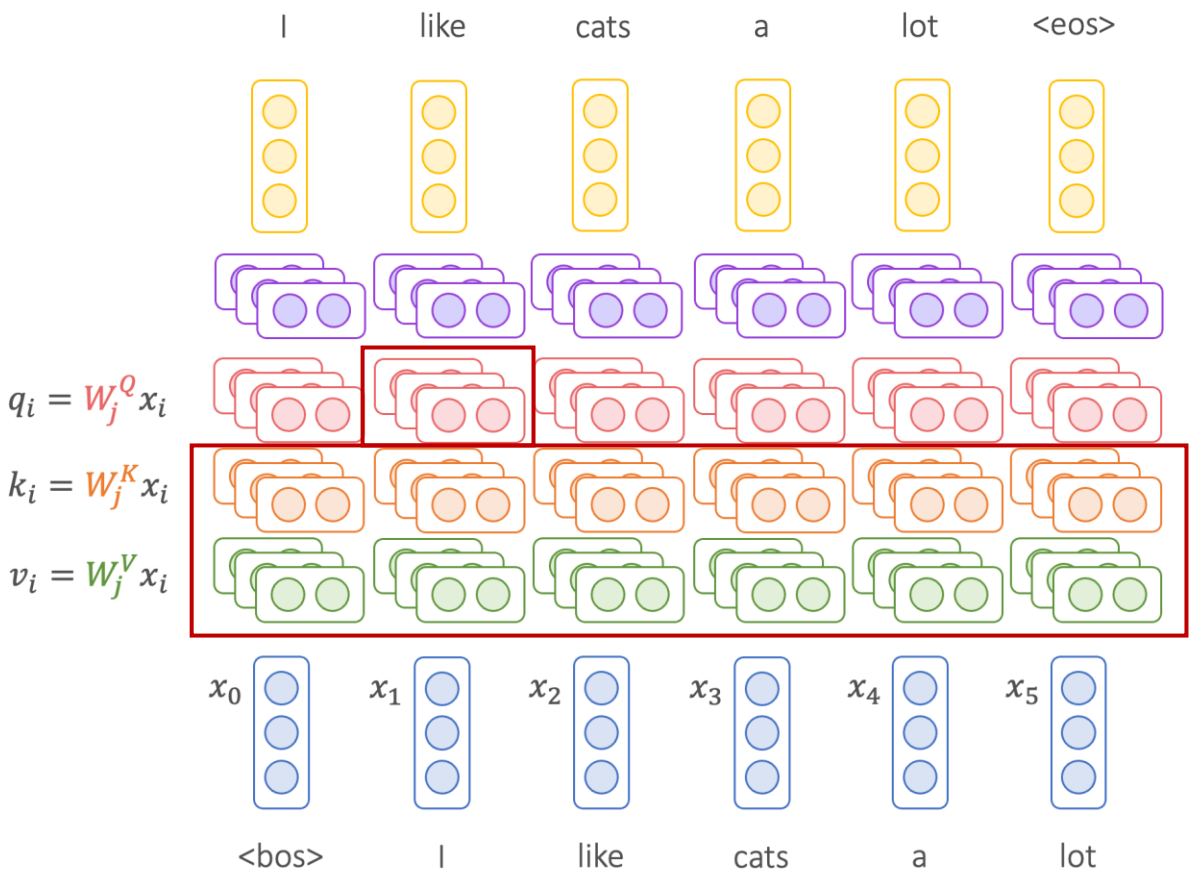


Transformer Encoder

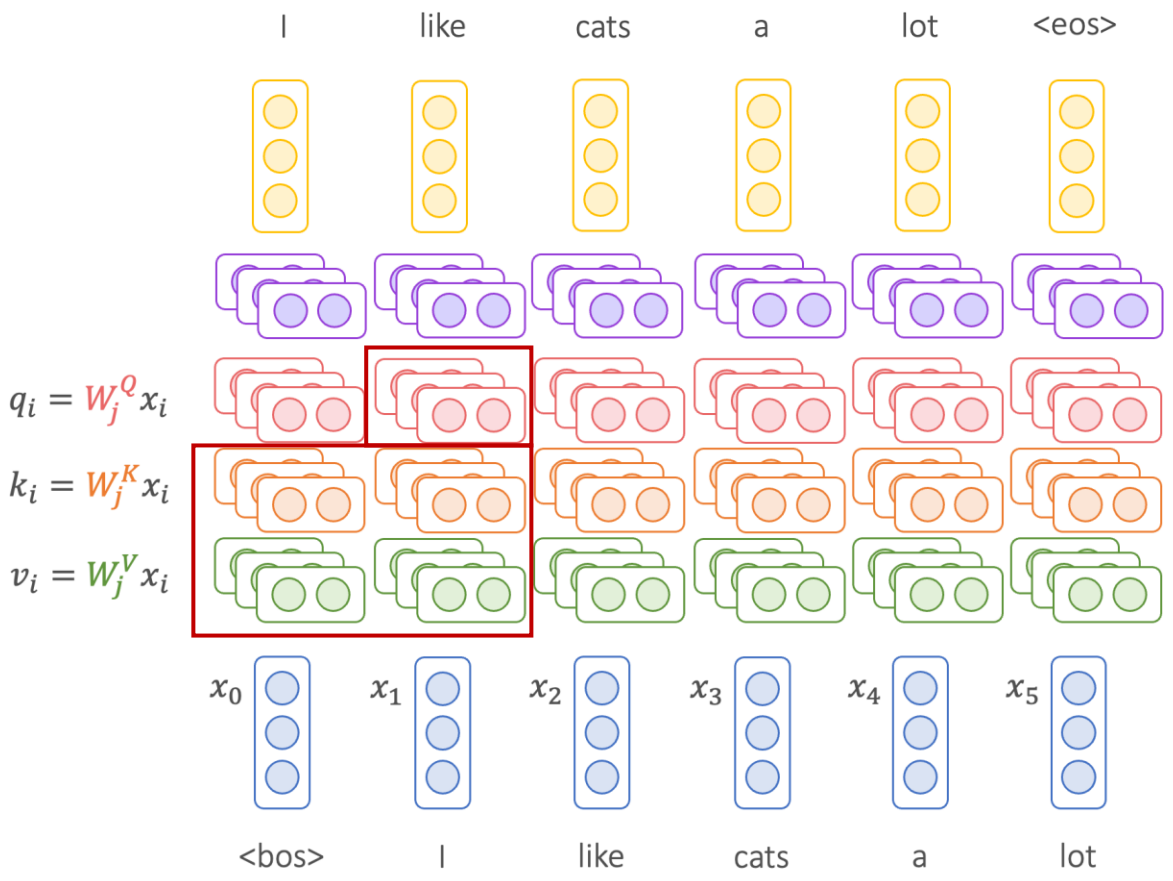


Transformer Decoder

Transformer Encoder vs. Transformer Decoder

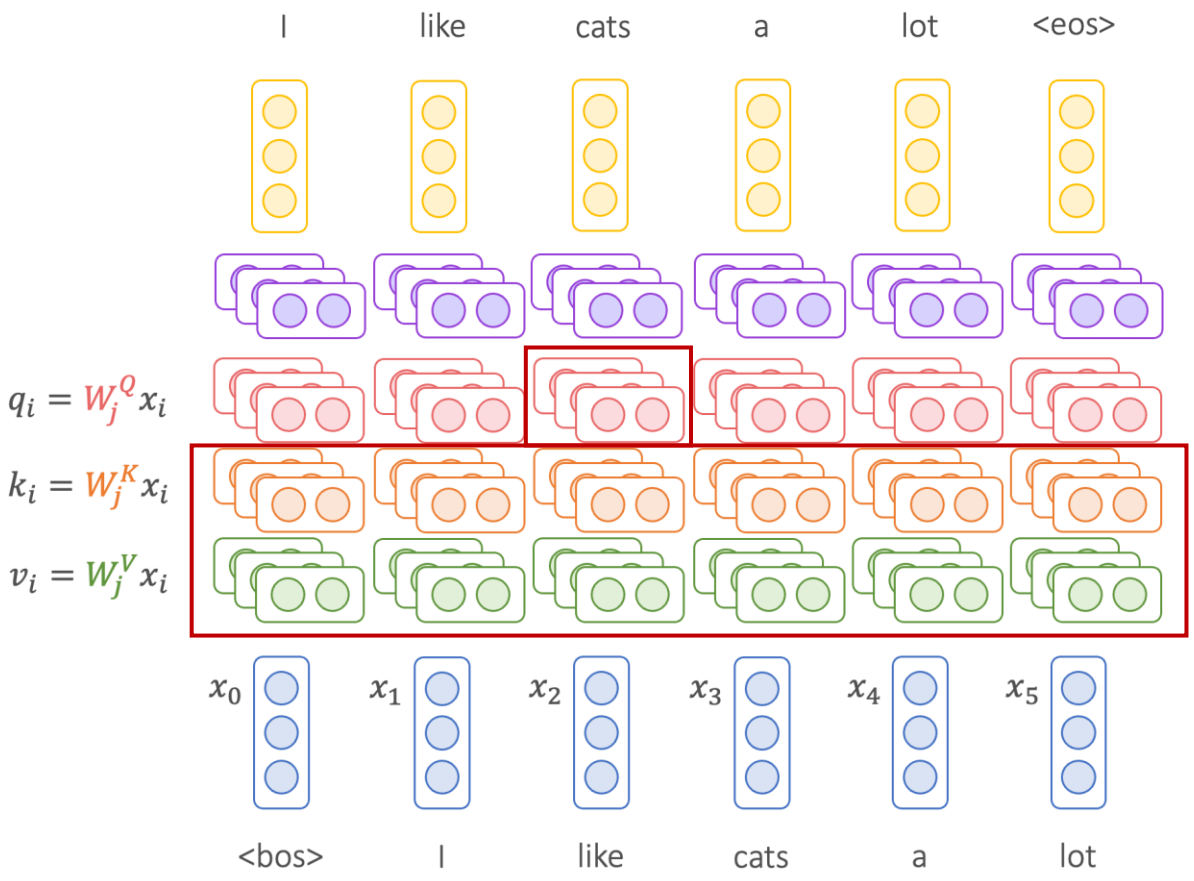


Transformer Encoder

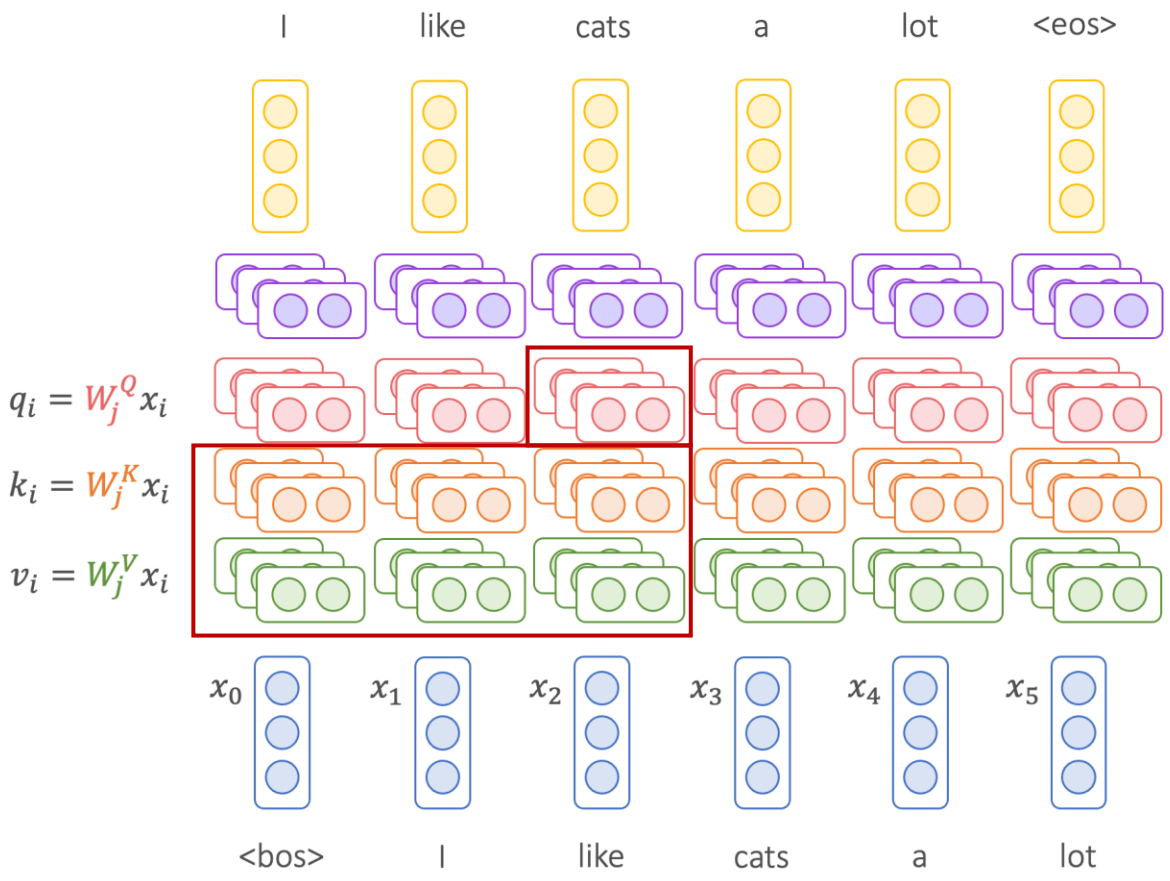


Transformer Decoder

Transformer Encoder vs. Transformer Decoder



Transformer Encoder

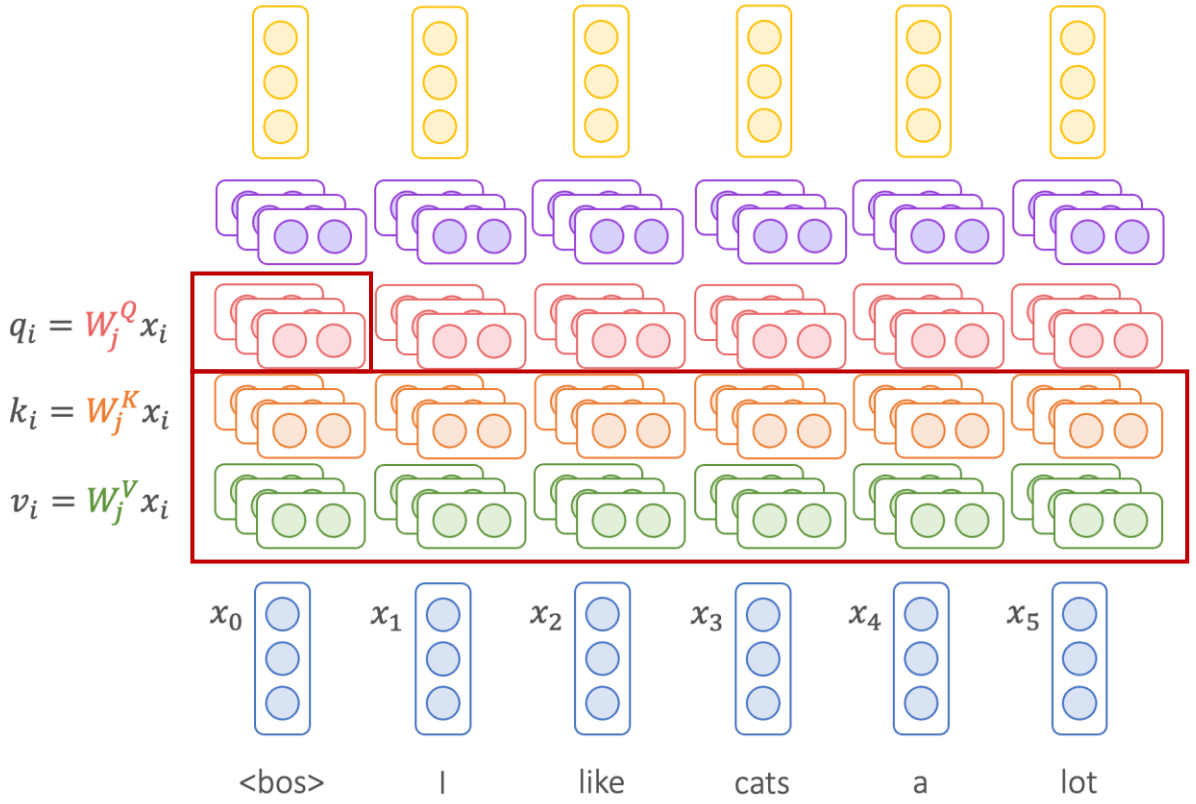


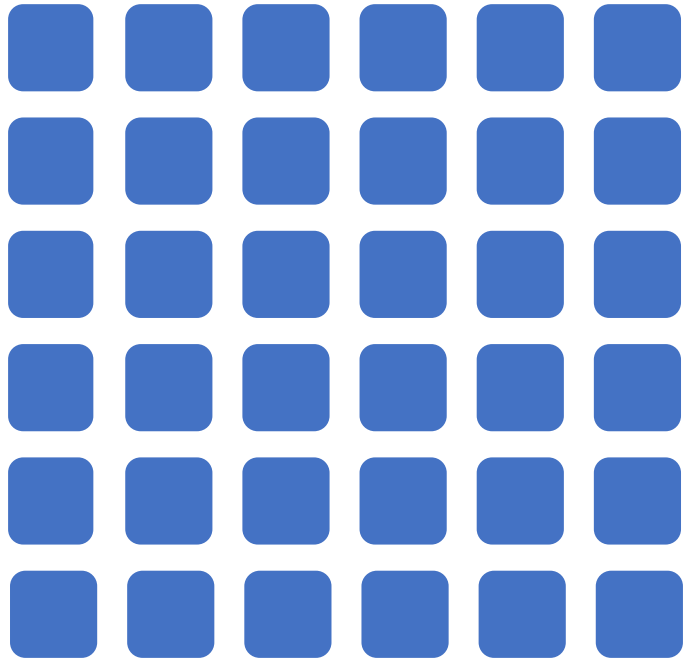
Transformer Decoder

Transformer Encoder vs. Transformer Decoder

- When computing attention for one word
  - Encoder: can see the words before and after this word
  - Decoder: can see the words only before this word

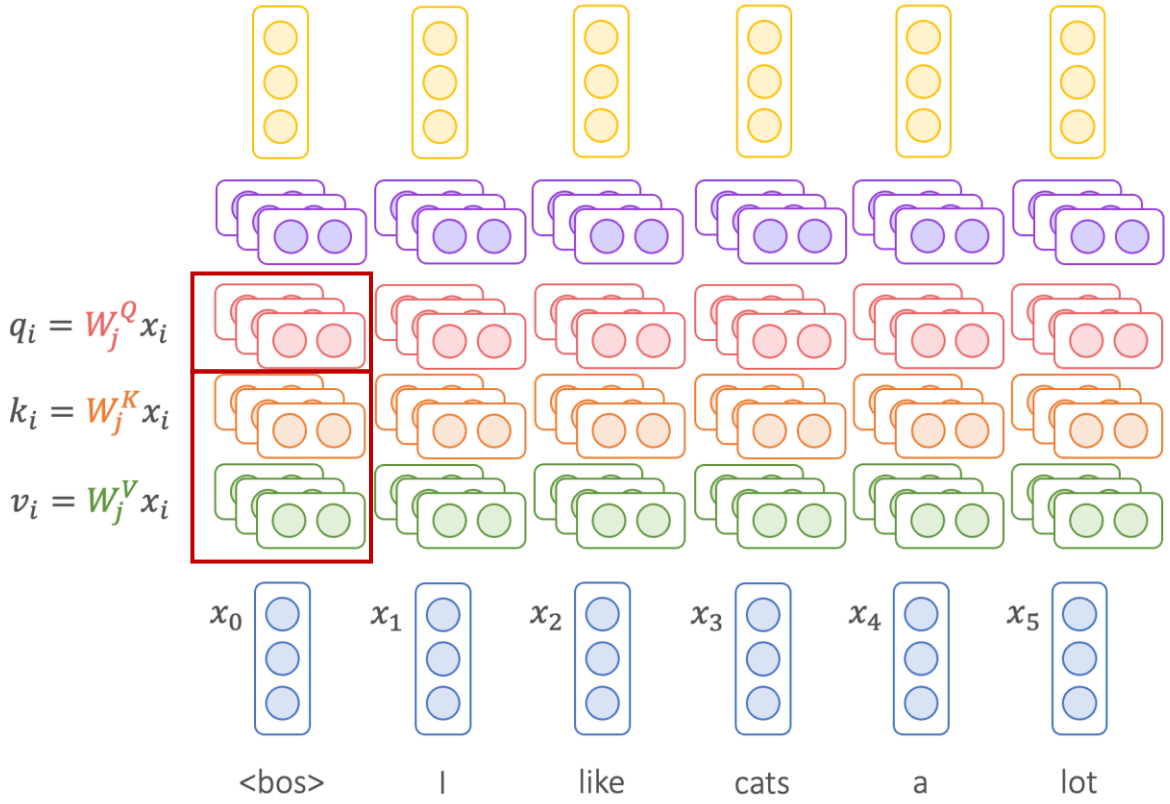
Masked Attention for Transformer Encoder

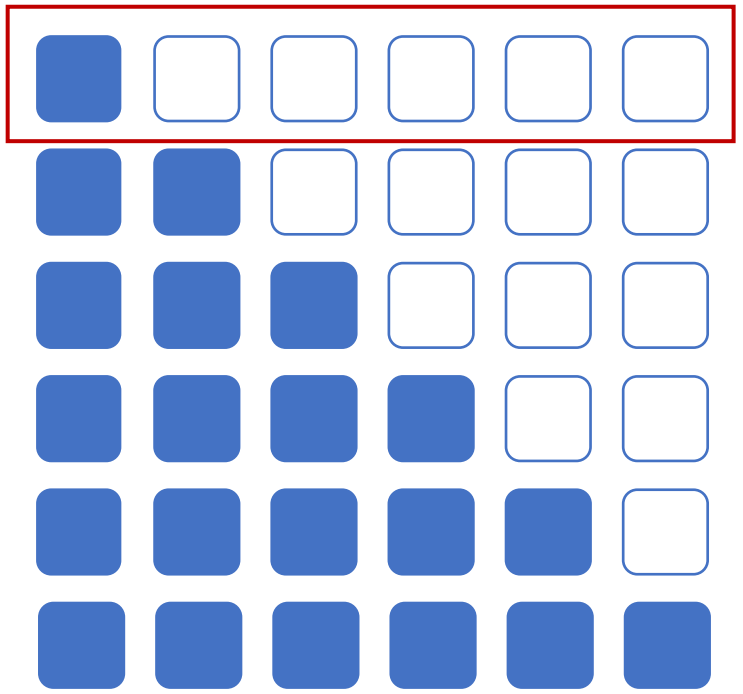




No Masking

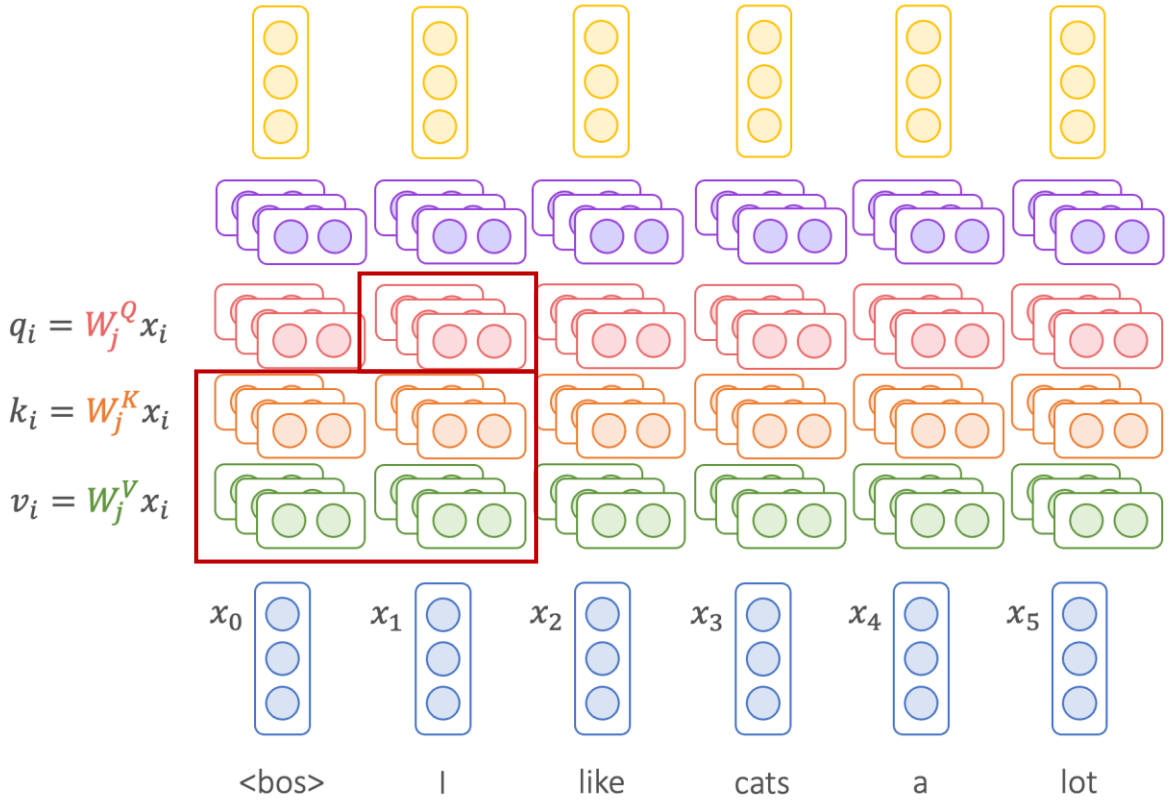
Masked Attention for Transformer Decoder

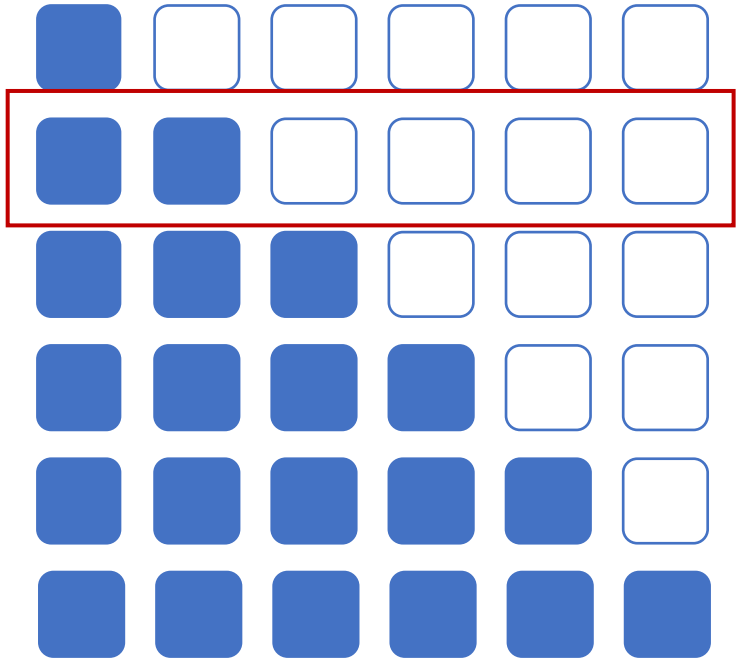




Causal Masking

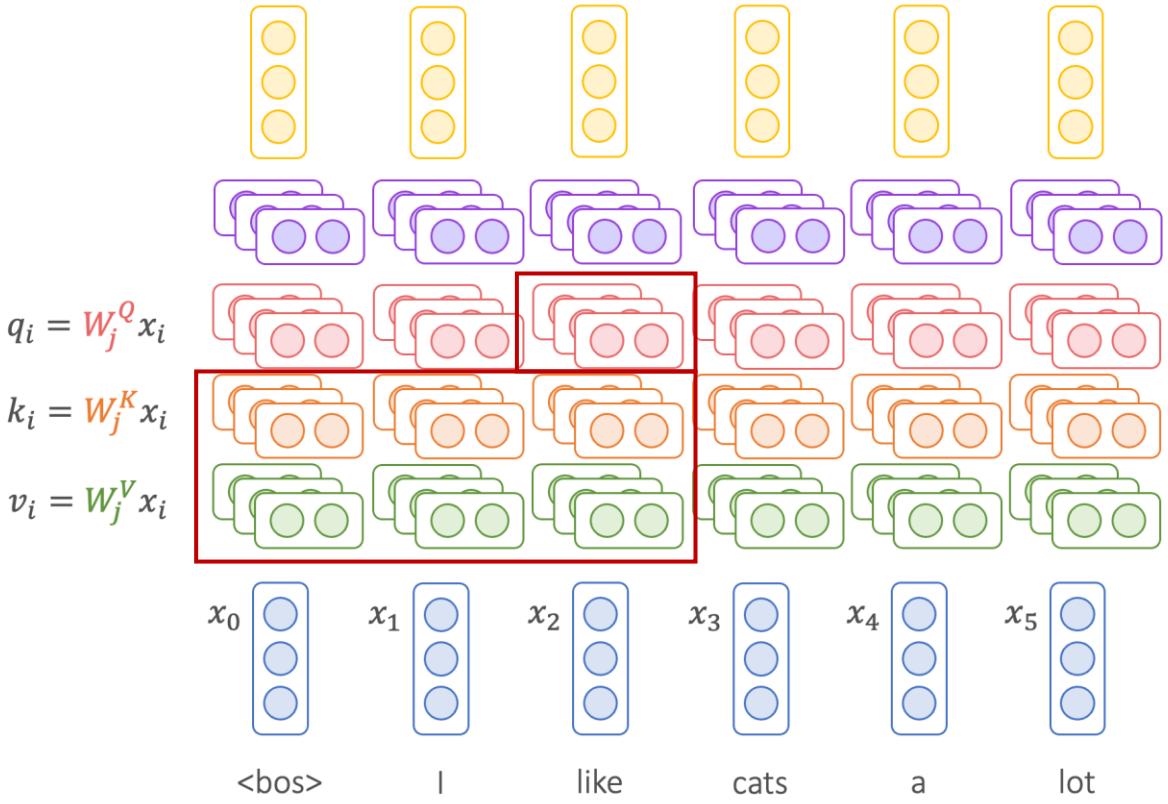
Masked Attention for Transformer Decoder

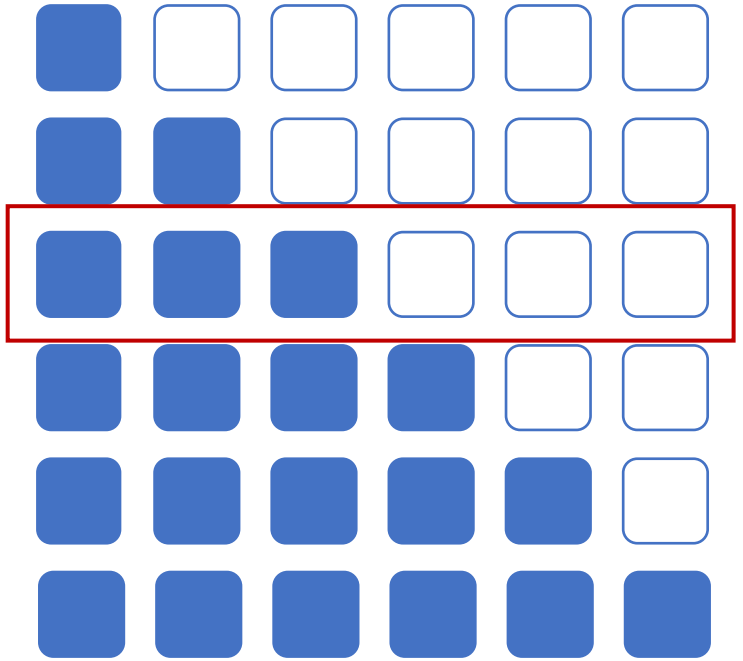




Causal Masking

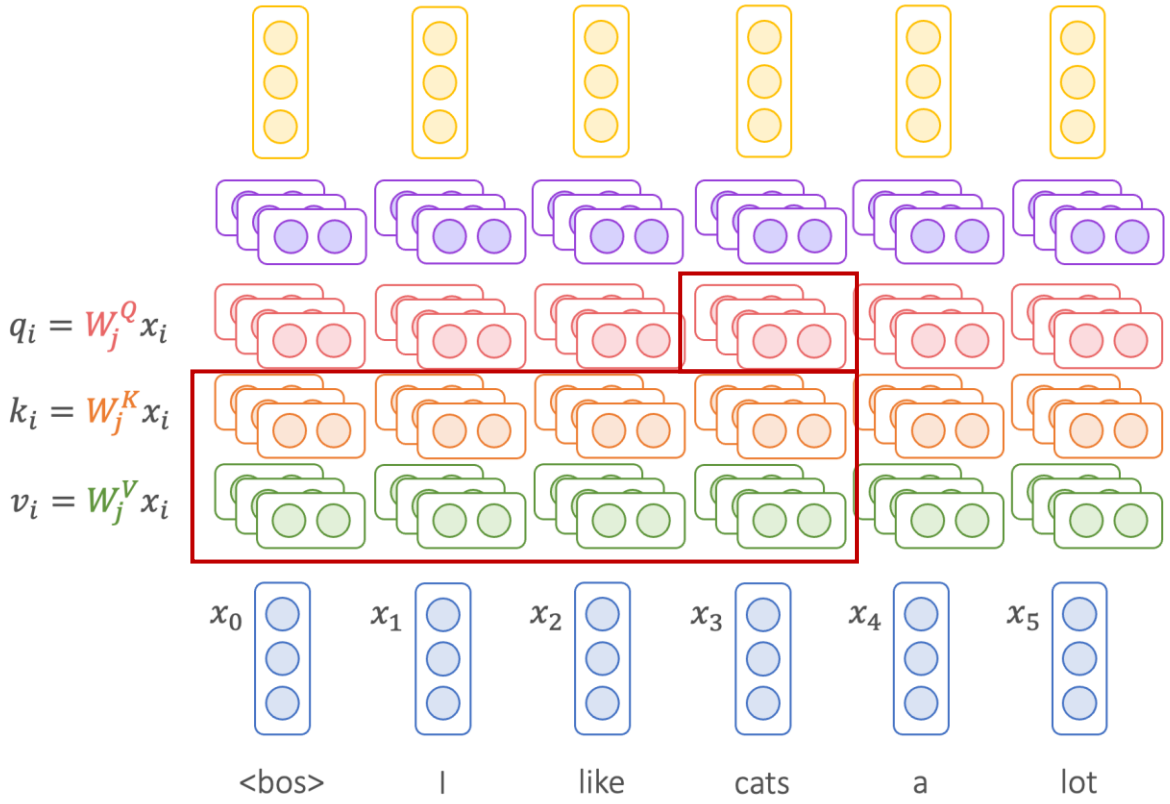
Masked Attention for Transformer Decoder

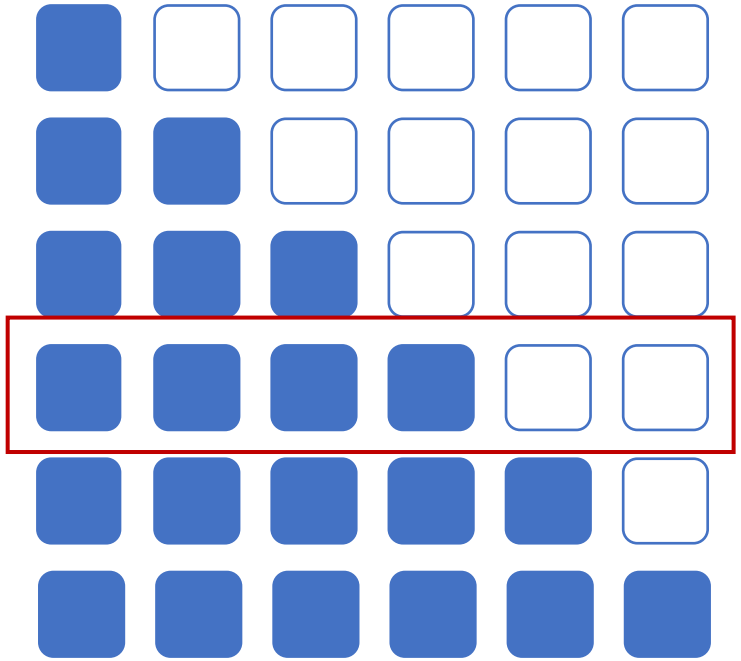




Causal Masking

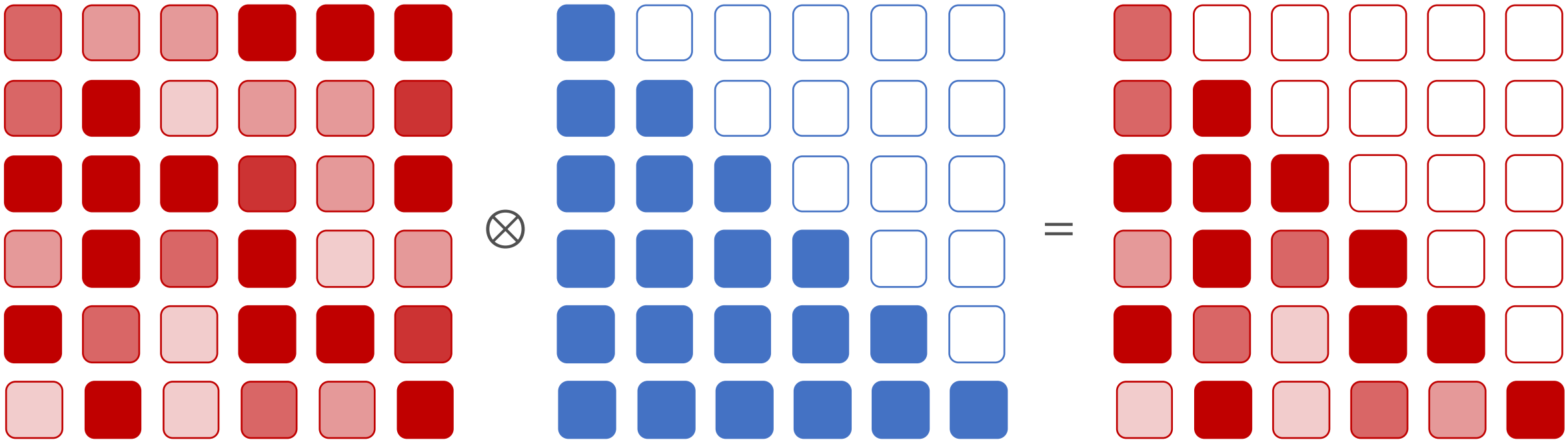
Masked Attention for Transformer Decoder





Causal Masking

Masked Attention: Implementation

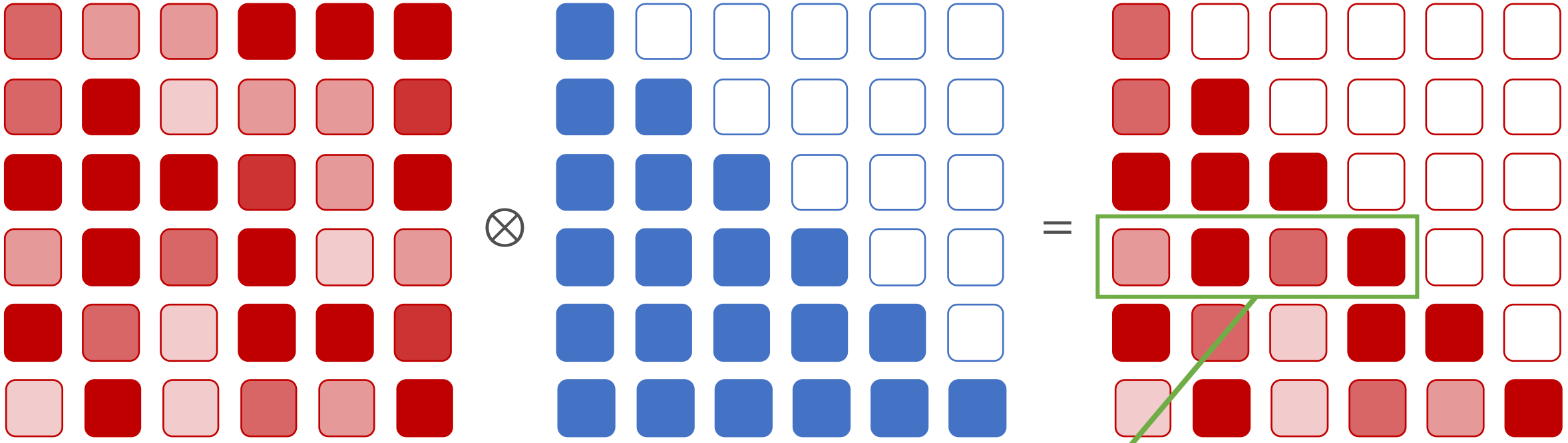


All-Pair Attention Scores

Causal Masking

Causal Attention Scores

Masked Attention: Implementation



All-Pair Attention Scores

Causal Masking

Causal Attention Scores

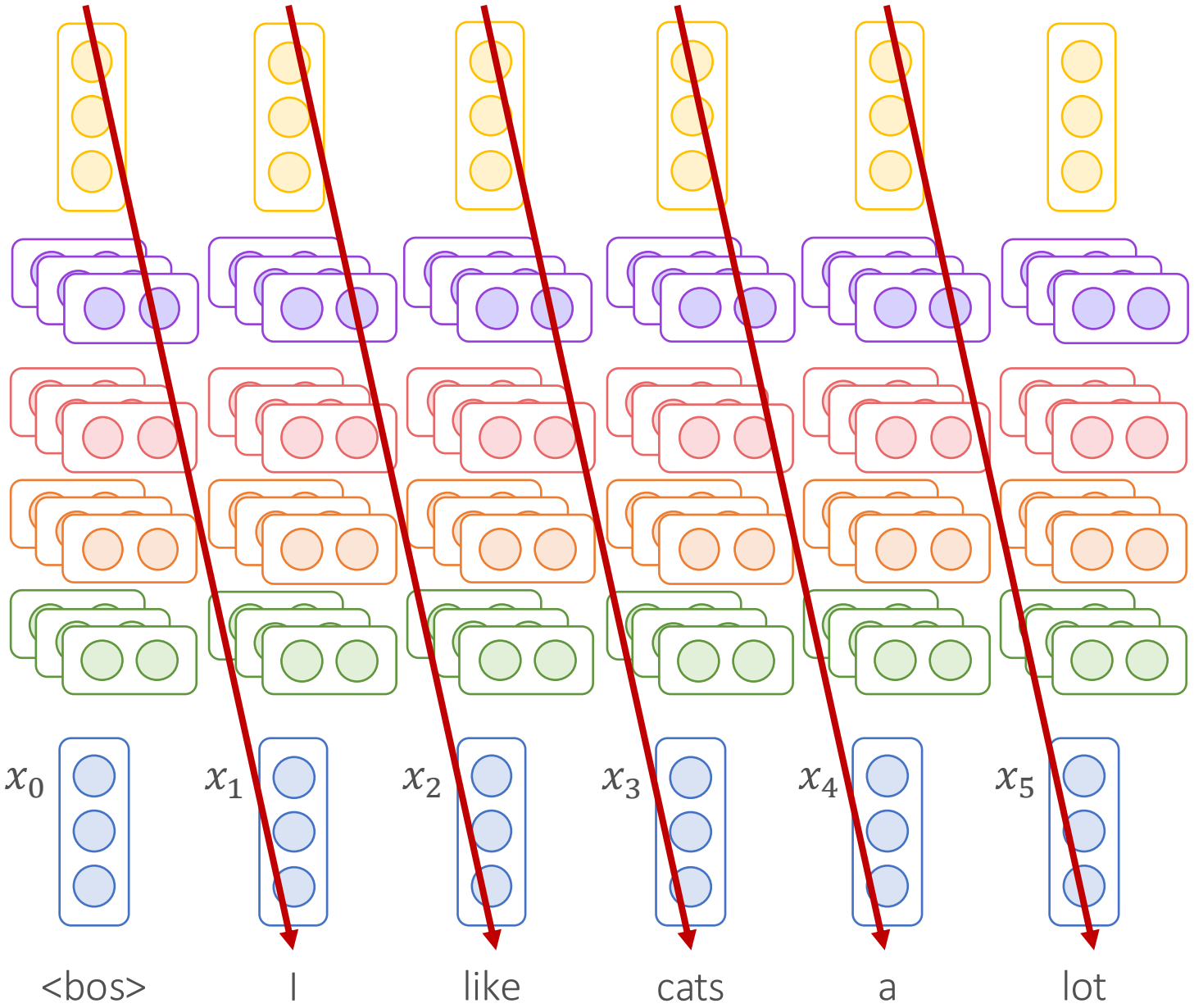
Normalize attention weights
& Weighted average value vectors

Transformer Decoder

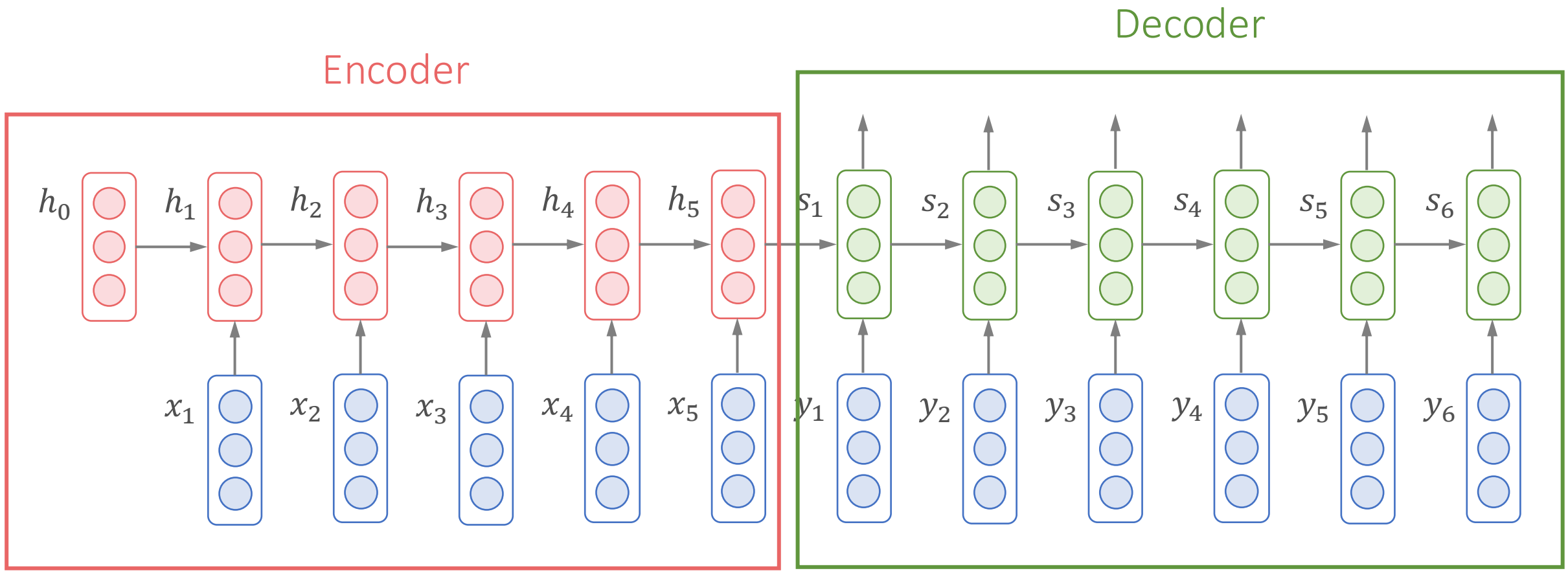
$$q_i = W_j^Q x_i$$

$$k_i = W_j^K x_i$$

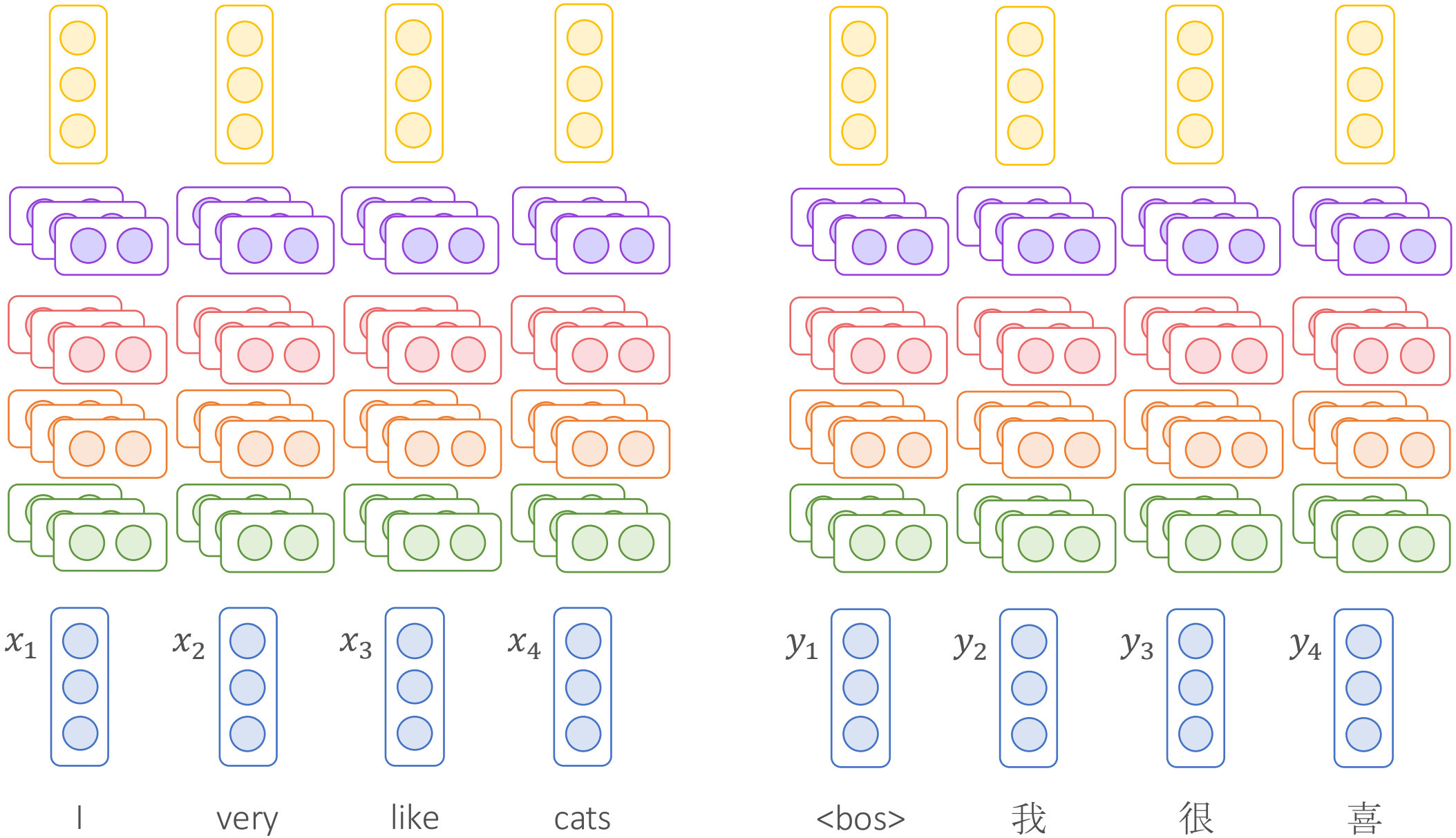
$$v_i = W_j^V x_i$$



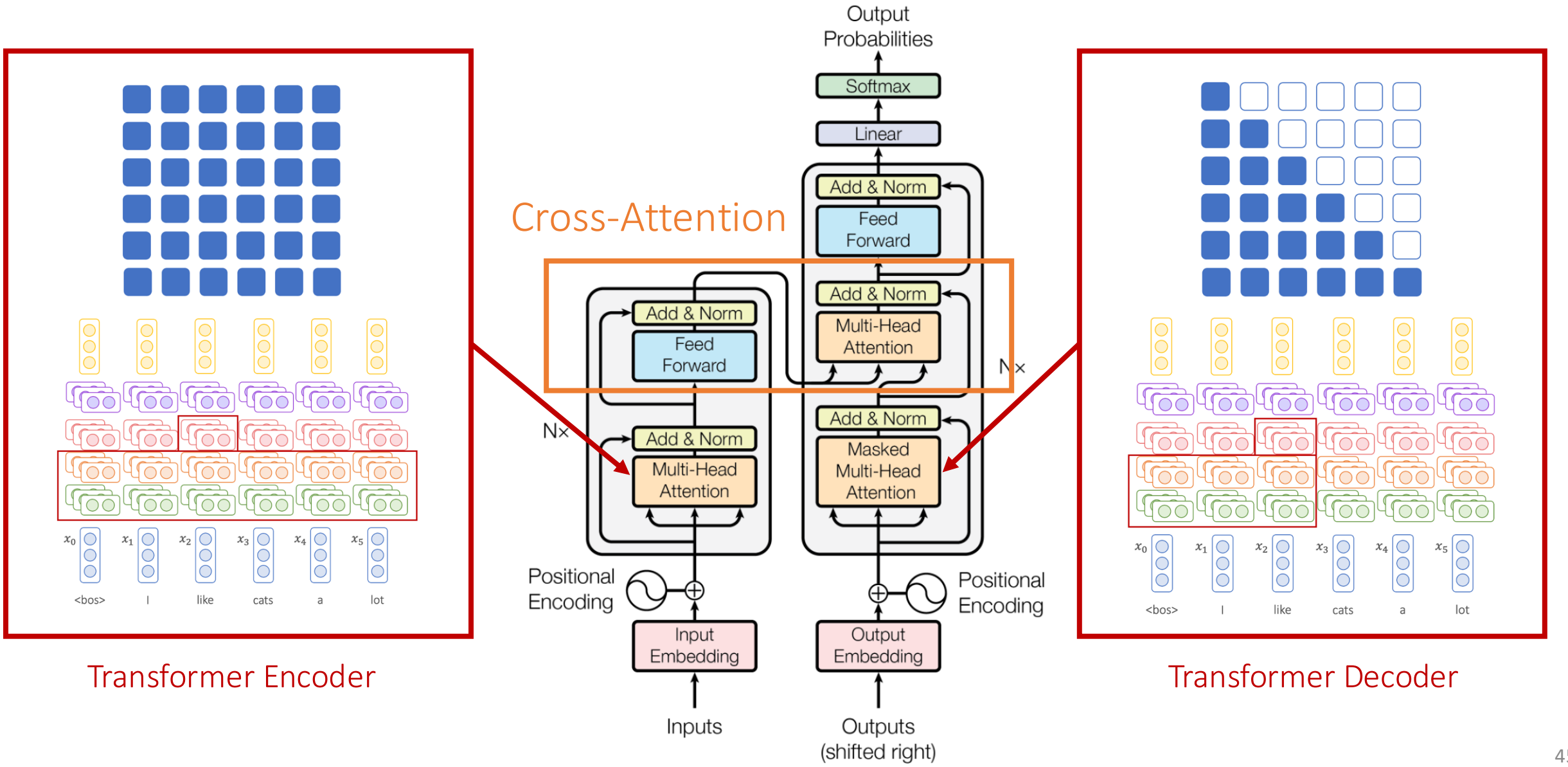
How About Encoder-Decoder (Sequence-to-Sequence)?



Transformer Encoder-Decoder (Sequence-to-Sequence)



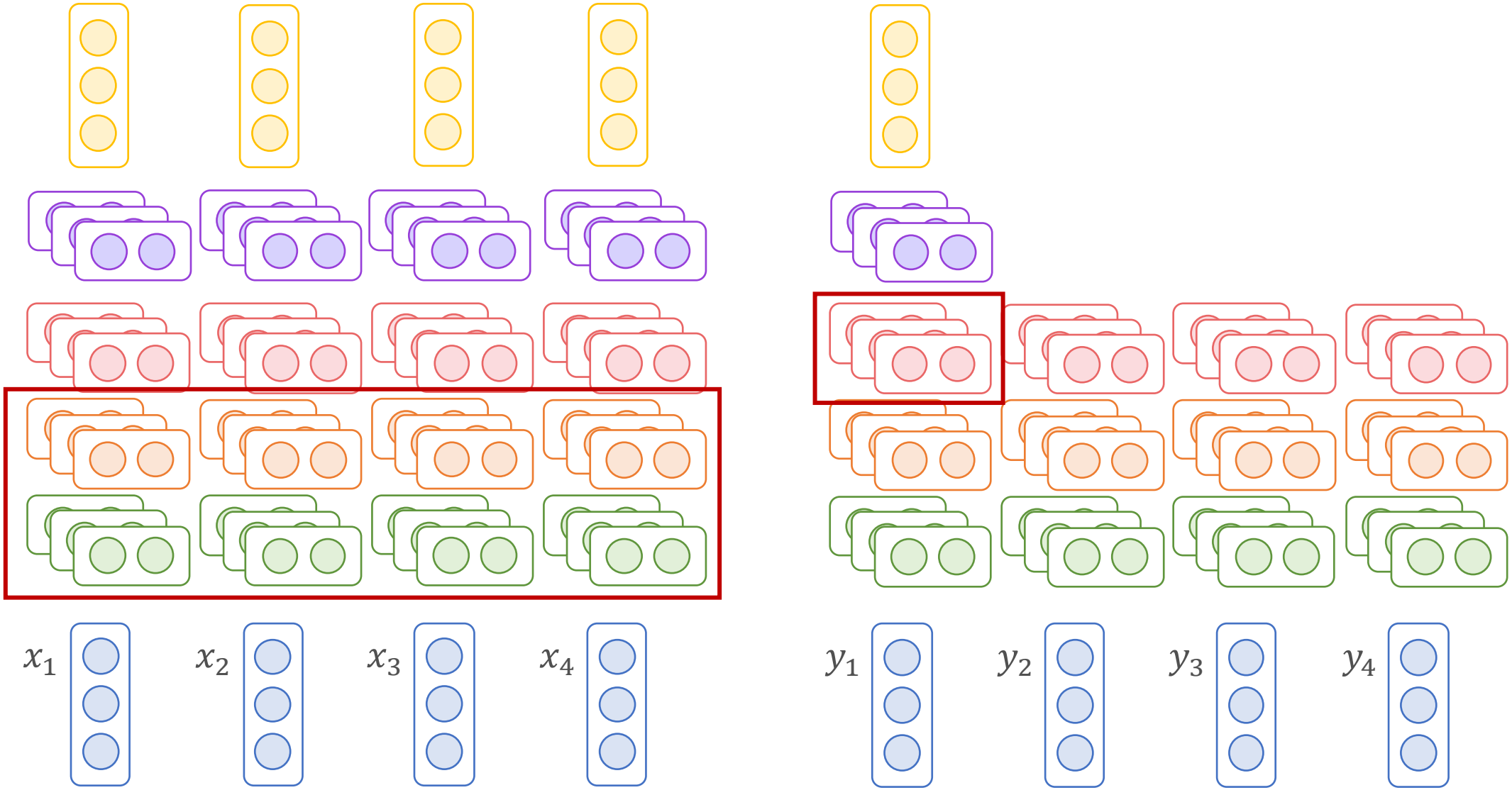
Transformer Encoder-Decoder (Sequence-to-Sequence)



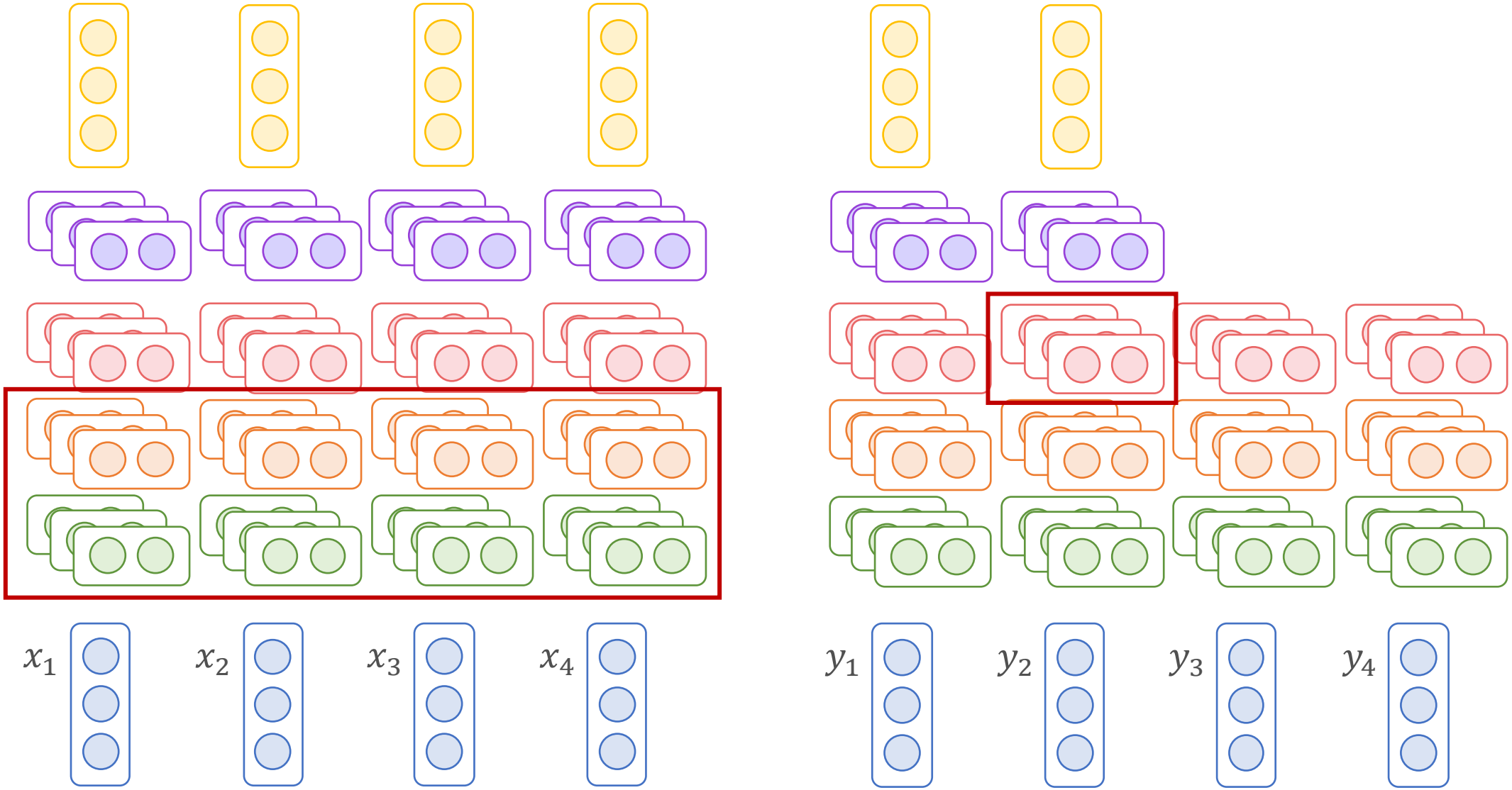
Transformer Encoder

Transformer Decoder

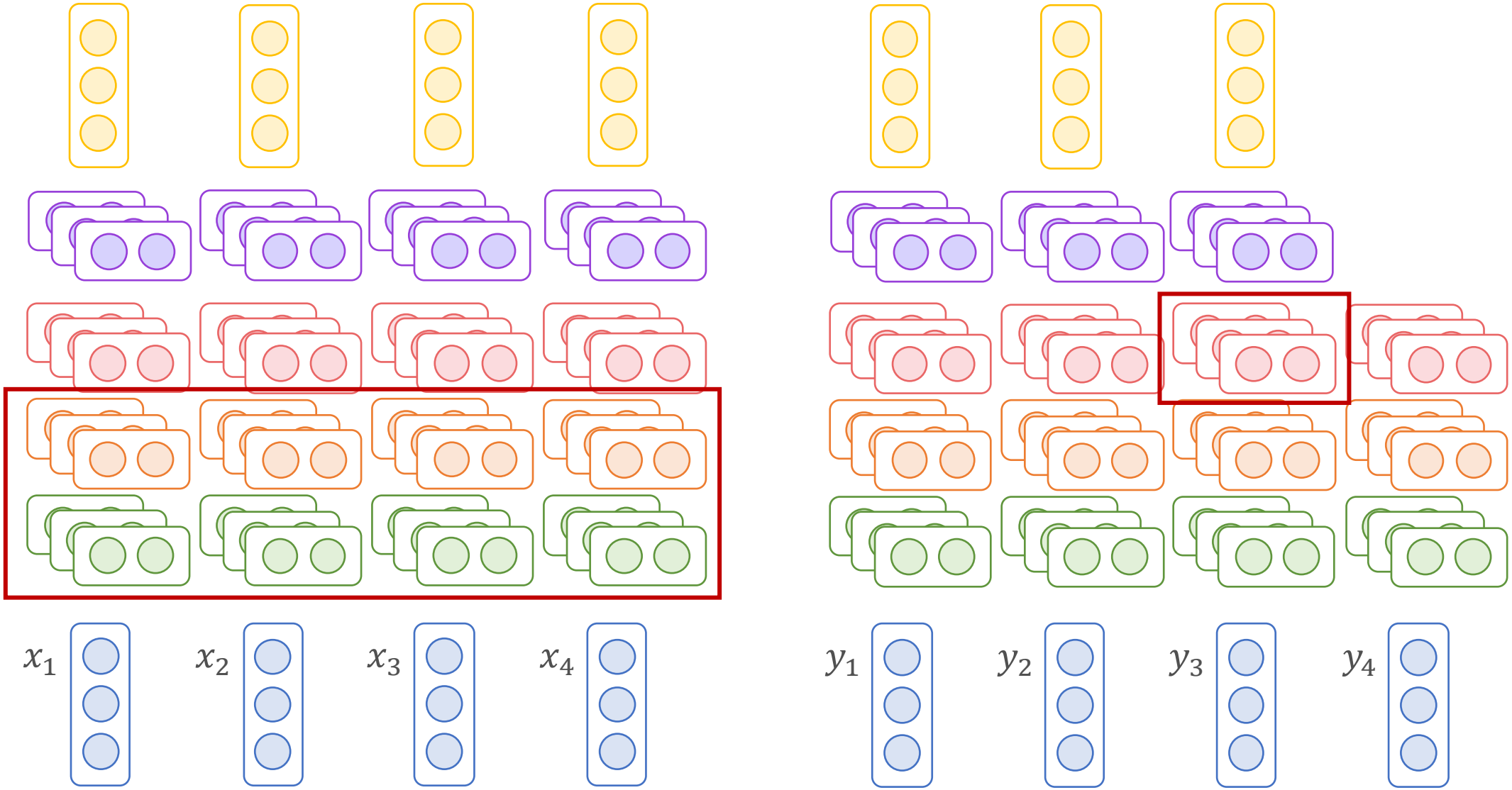
Cross-Attention



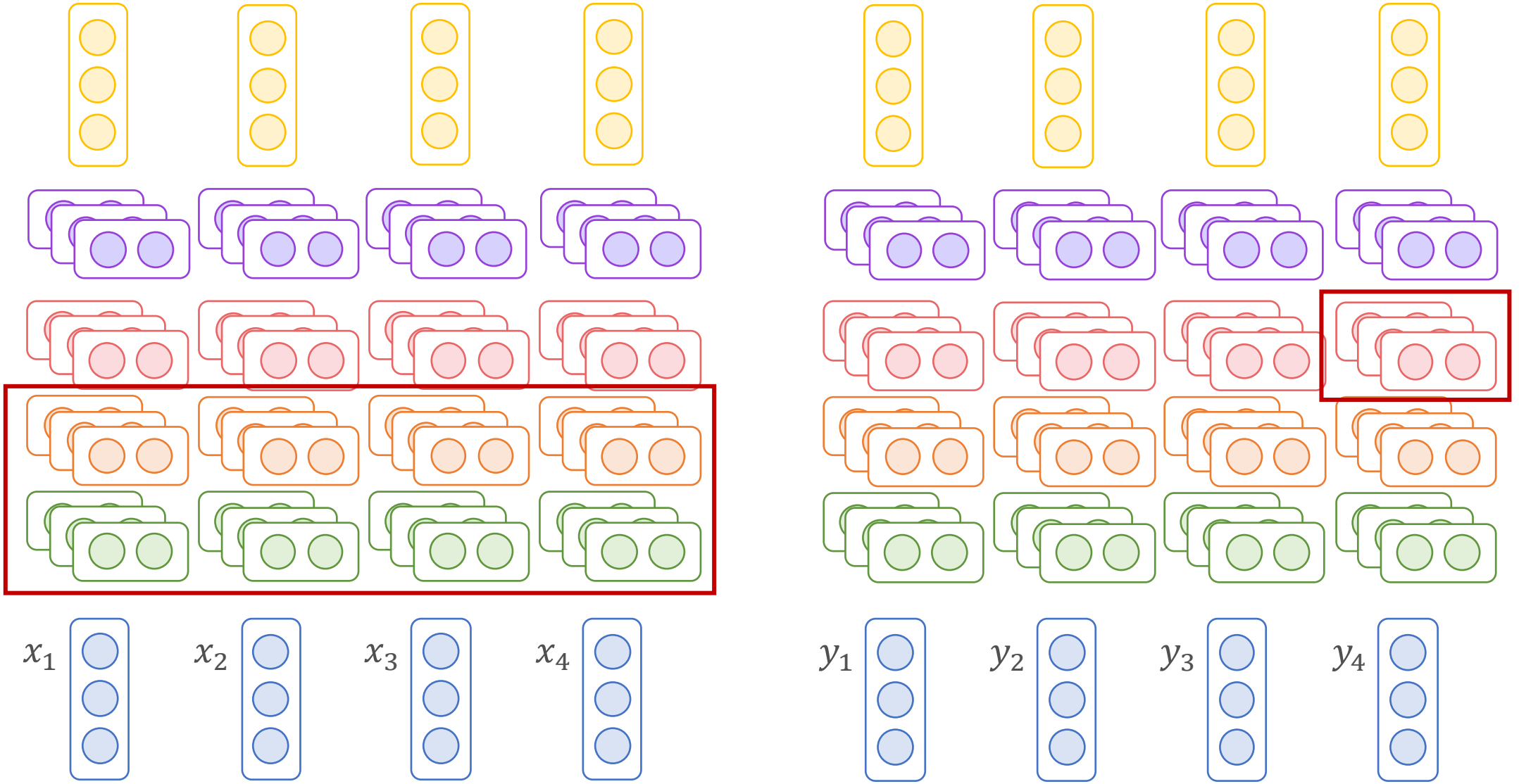
Cross-Attention



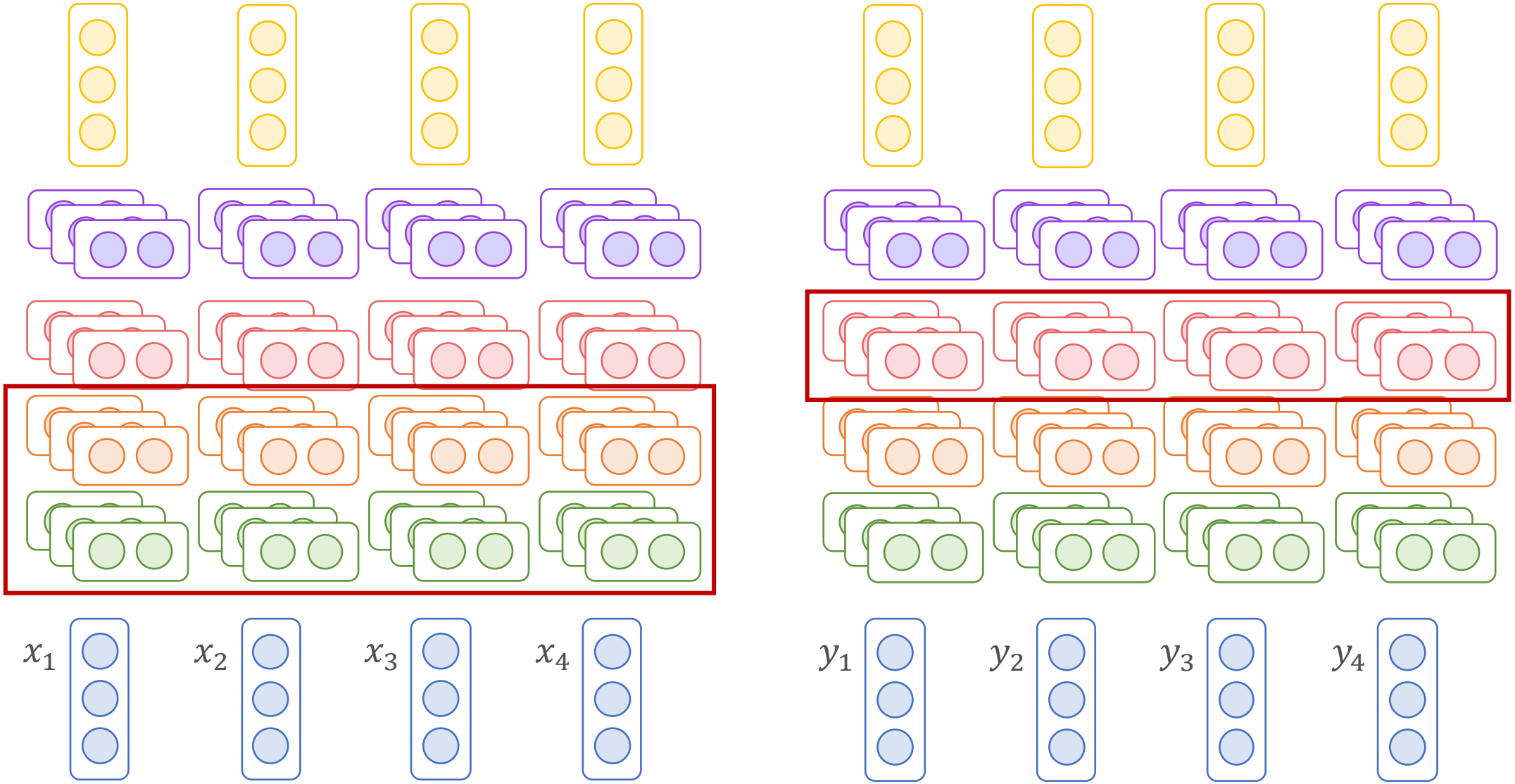
Cross-Attention



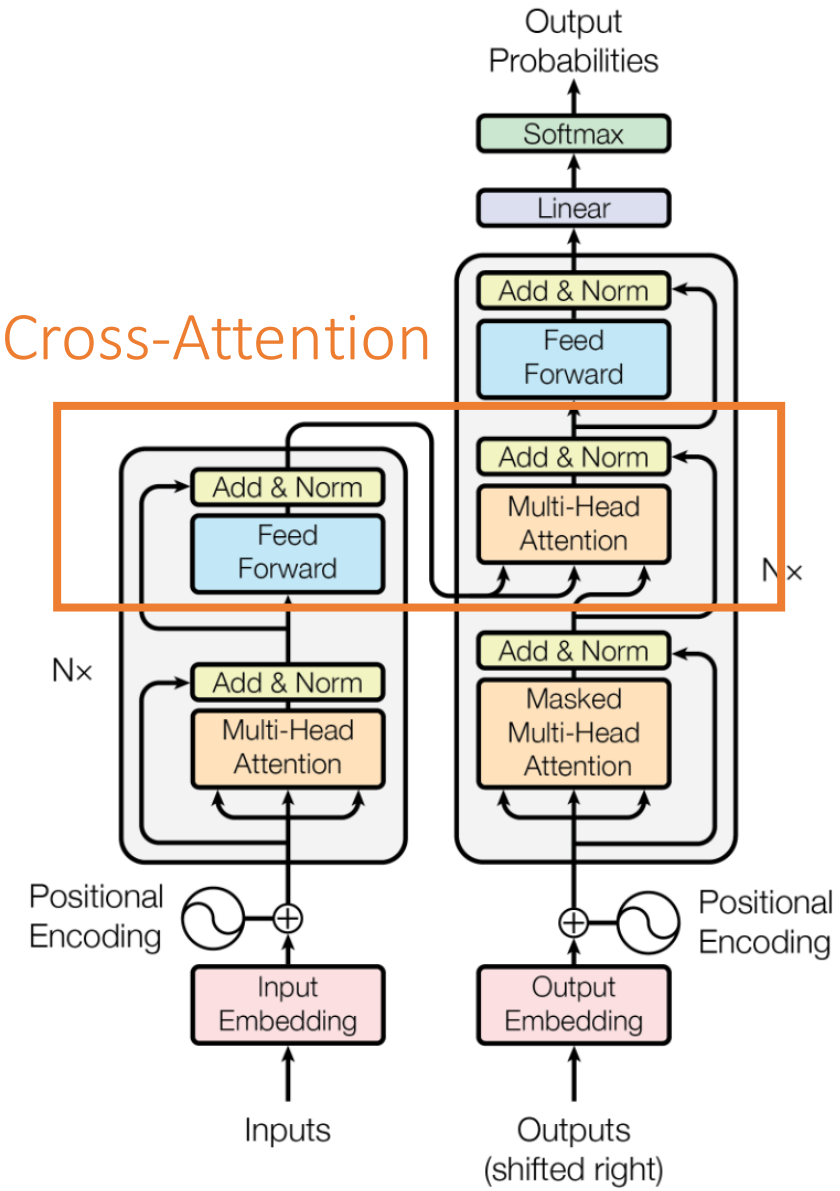
Cross-Attention



Cross-Attention



Transformer



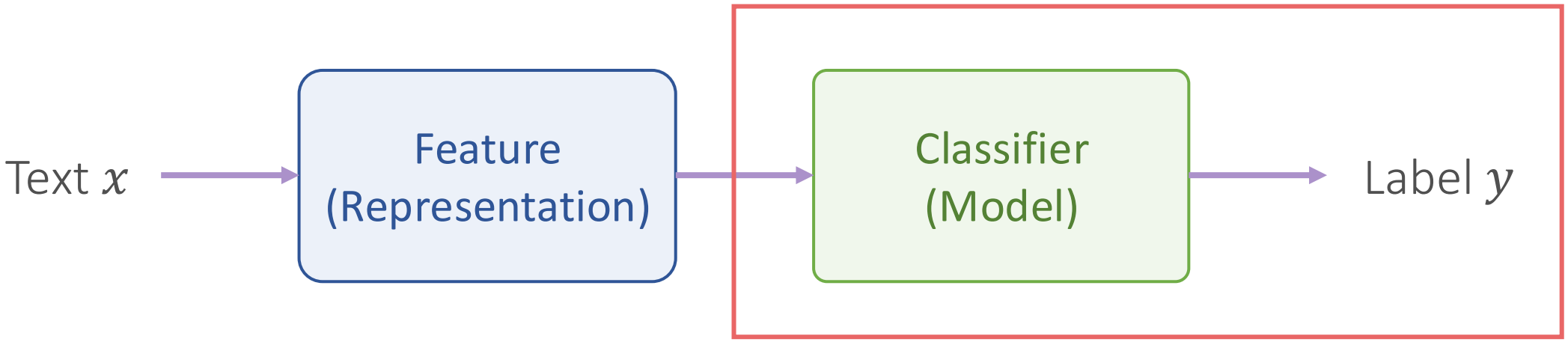
Cross-Attention

Transformer on Machine Translation

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Attention Is All You Need, 2017

A General Framework for Text Classification



- Teach the model how to make prediction y
- Logistic regression, neural networks, CNN, RNN, LSTM, Transformers

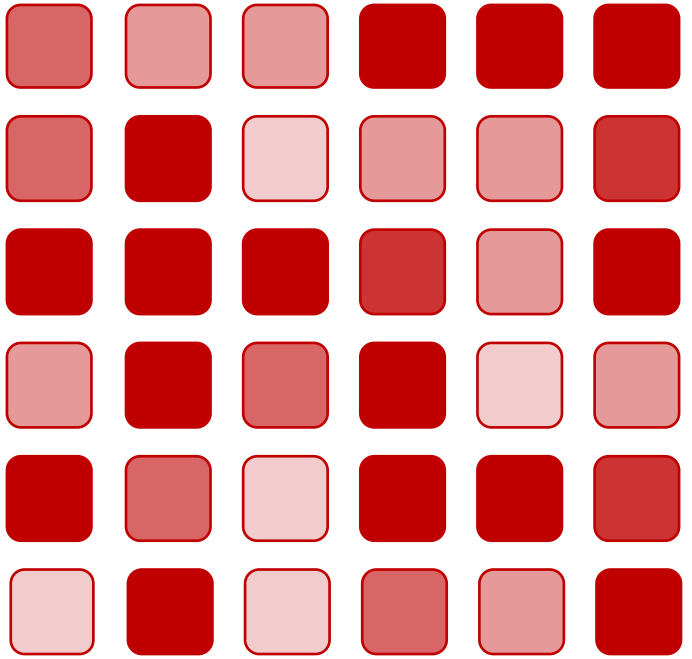
Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$

Lecture Plan

- Transformers
  - Encoder
  - Decoder
  - Encoder-Decoder
- Transformers Variants
  - Longformer
  - Relative Positional Encoding
  - RoFormer

Computation in Transformer

- All-pair attention scores
  - Complexity $O(length^2)$
- When the input is long → slow

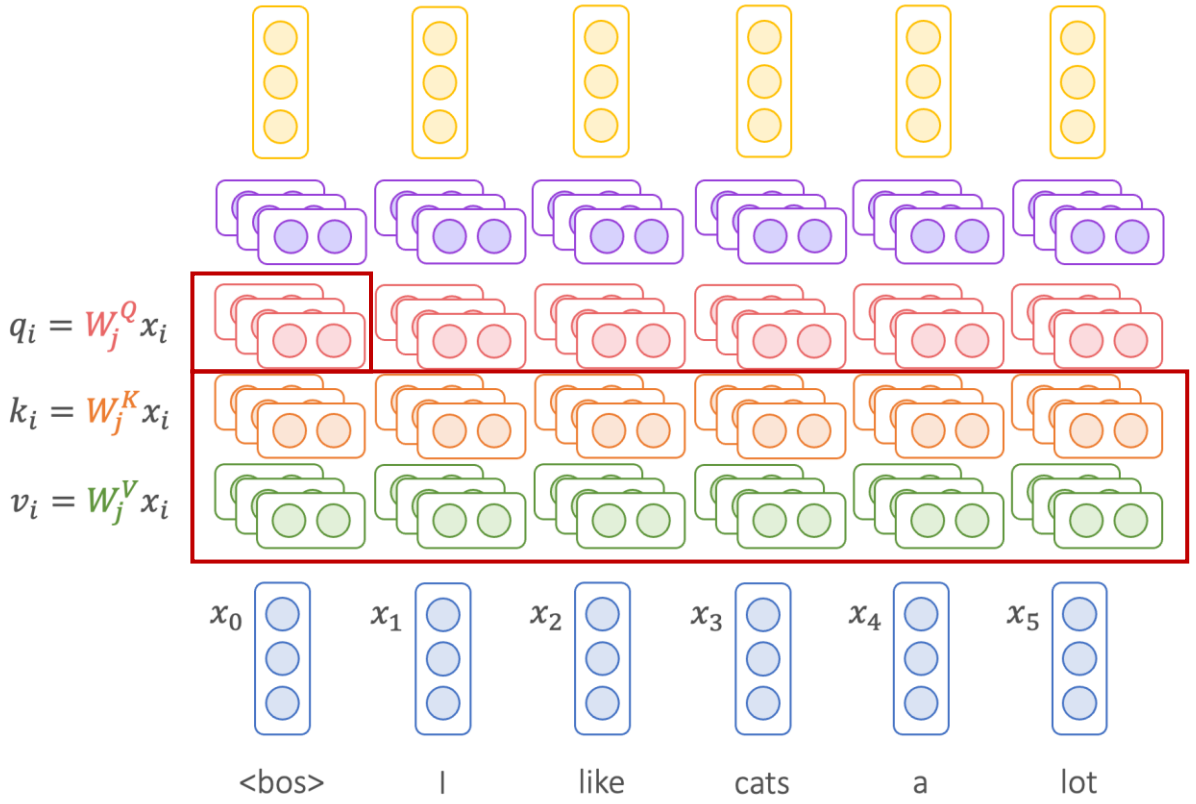


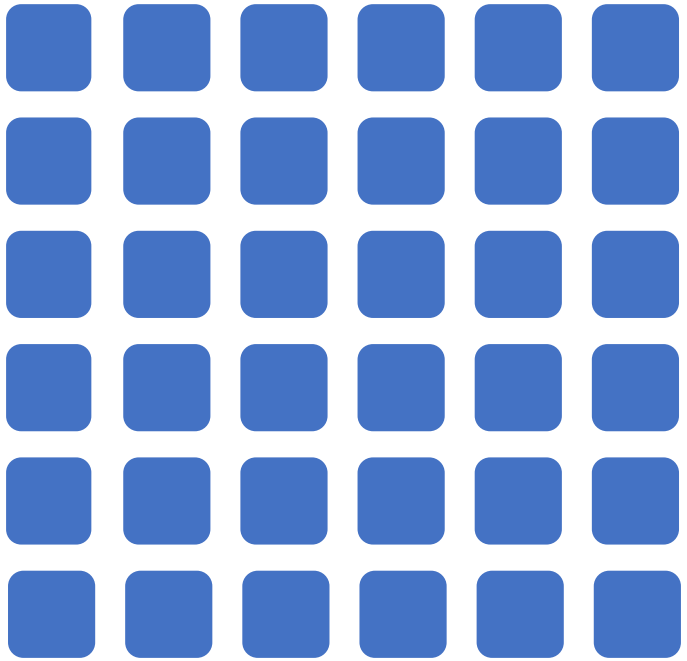
LongFormer

- Don’t compute all-pair attention score
  - Manipulate attention mask
- Capture local information to reduce computational load
  - Idea is similar to convolutional neural network

Longformer: The Long-Document Transformer, 2020

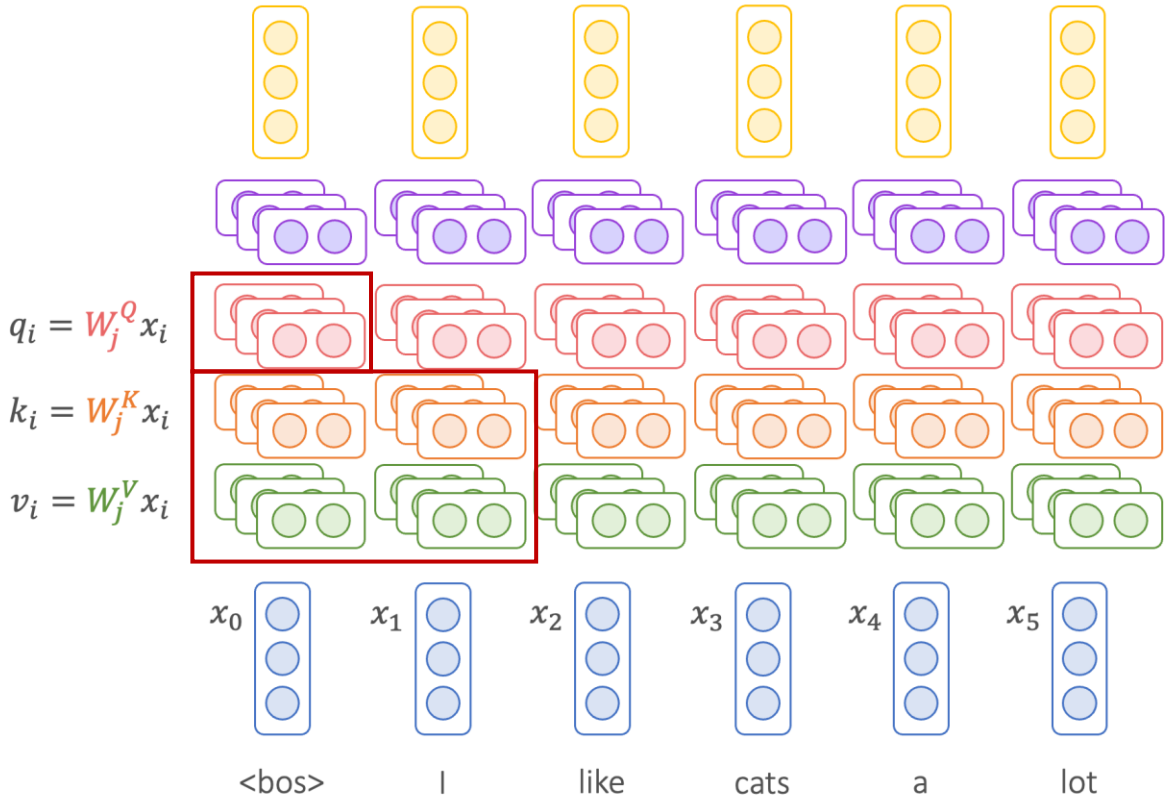
Transformer Encoder

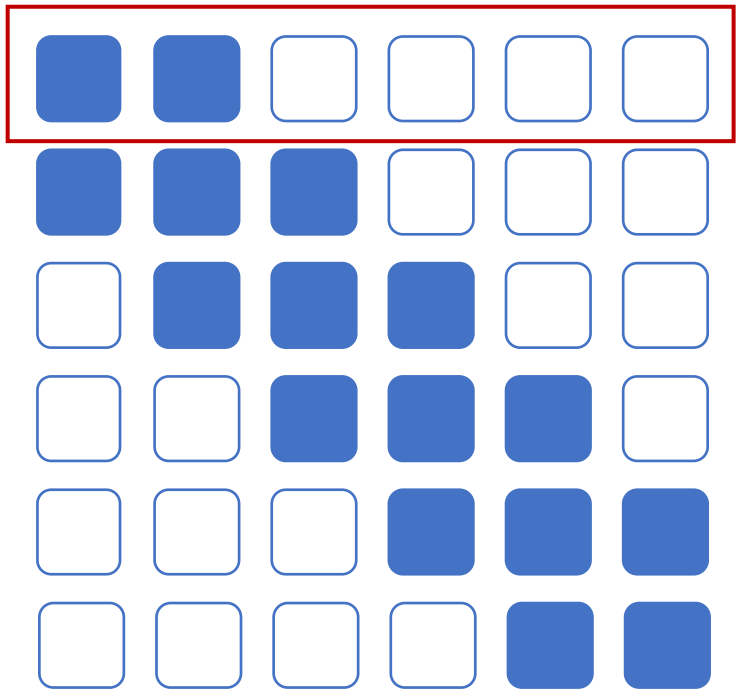




No Masking

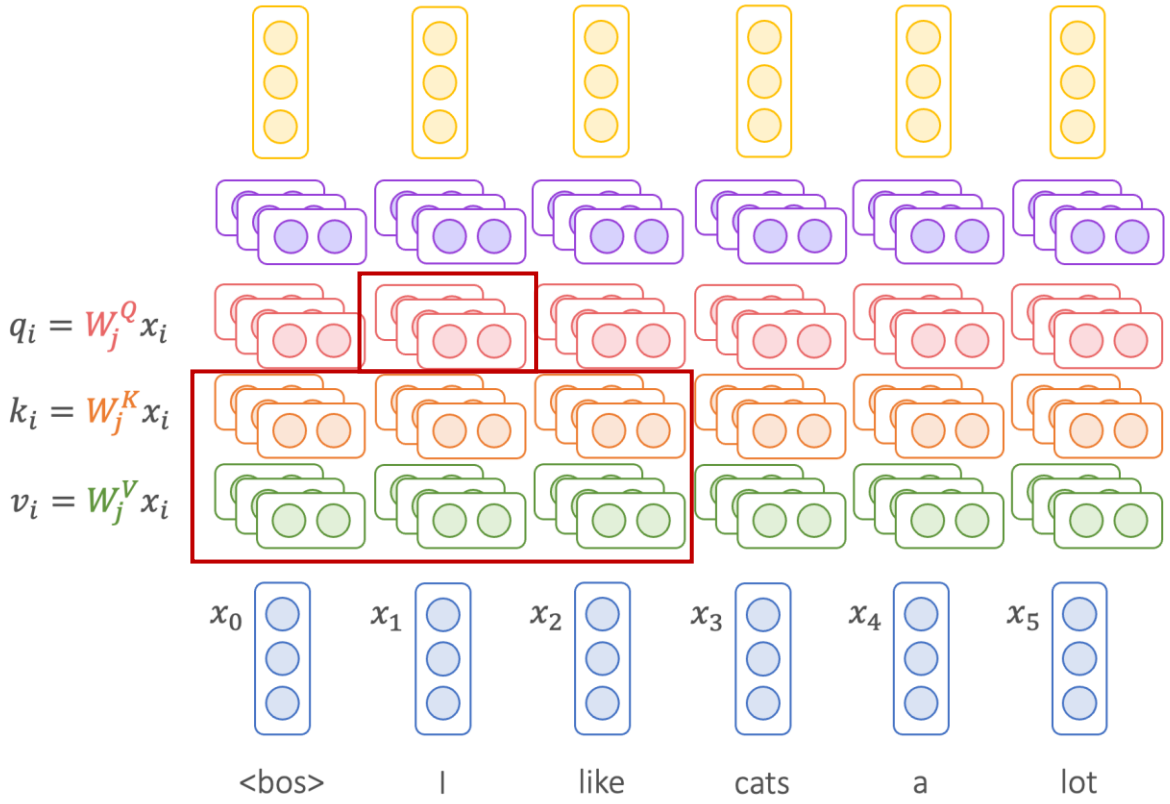
Sliding Window Attention Masking

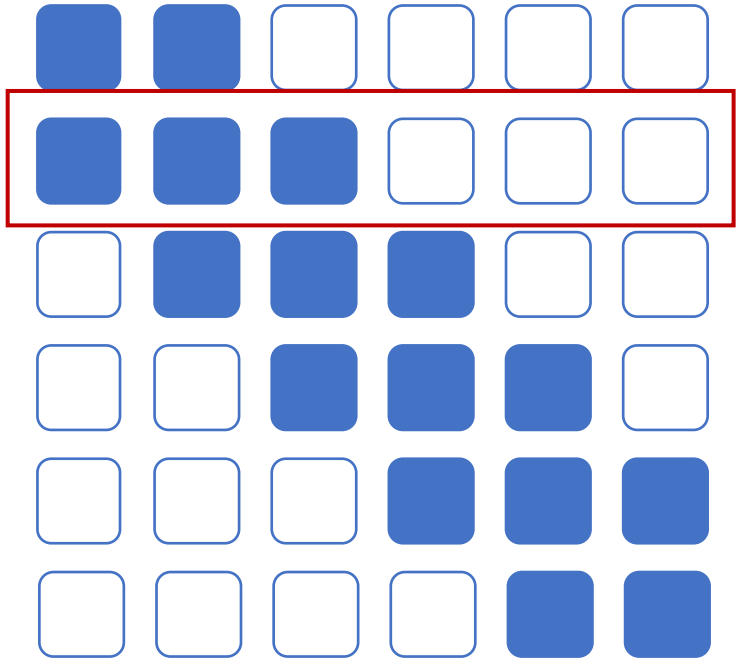




Sliding Window Attention Masking

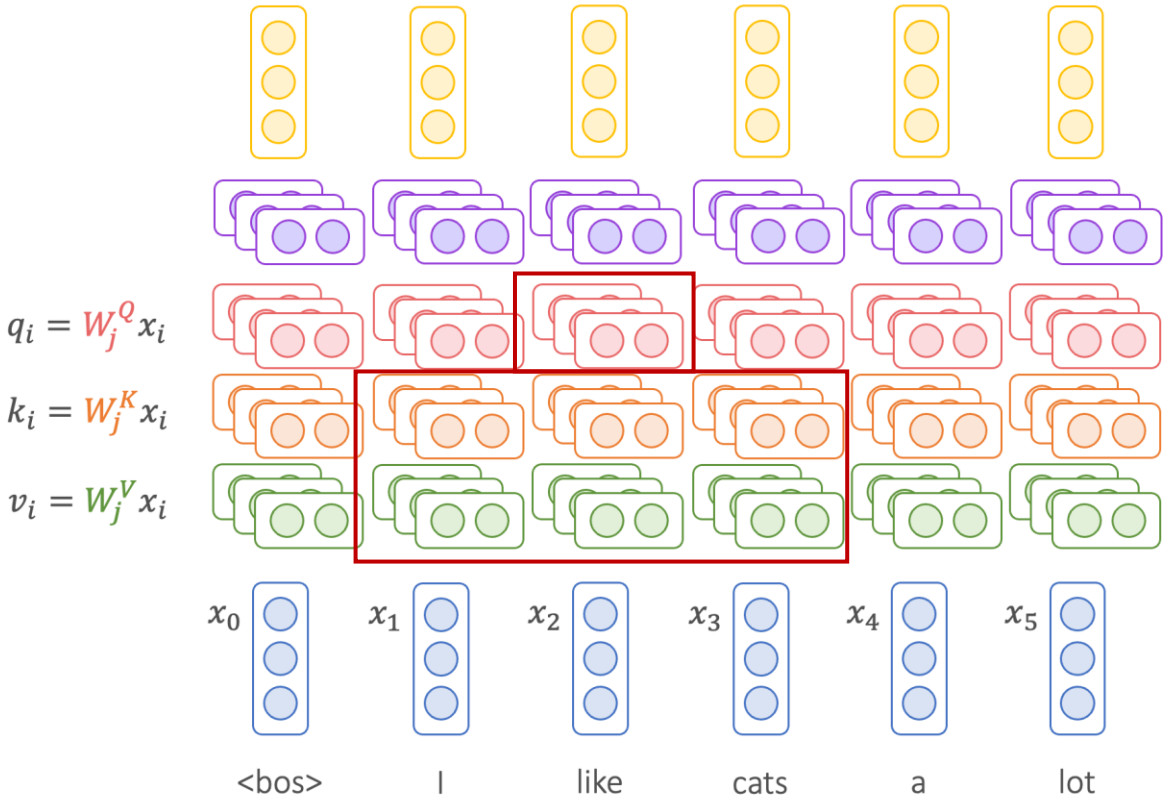
Sliding Window Attention Masking

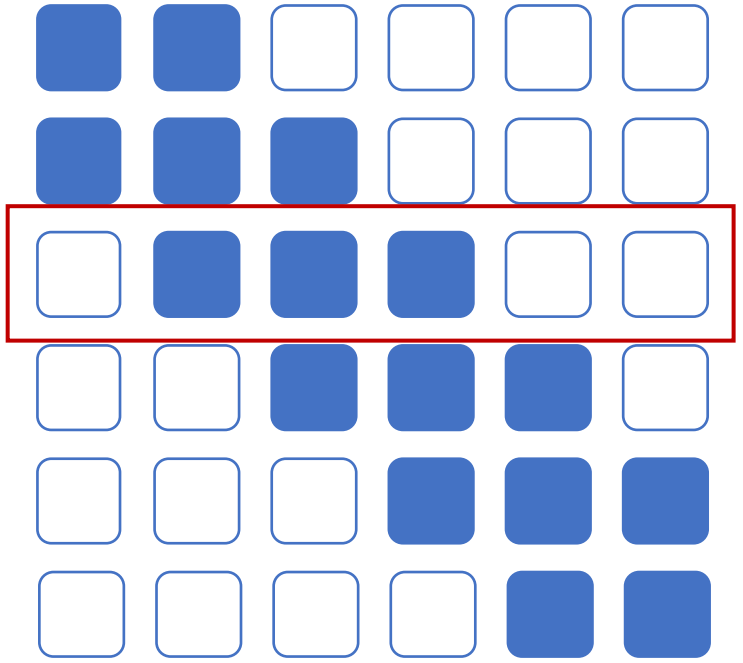




Sliding Window Attention Masking

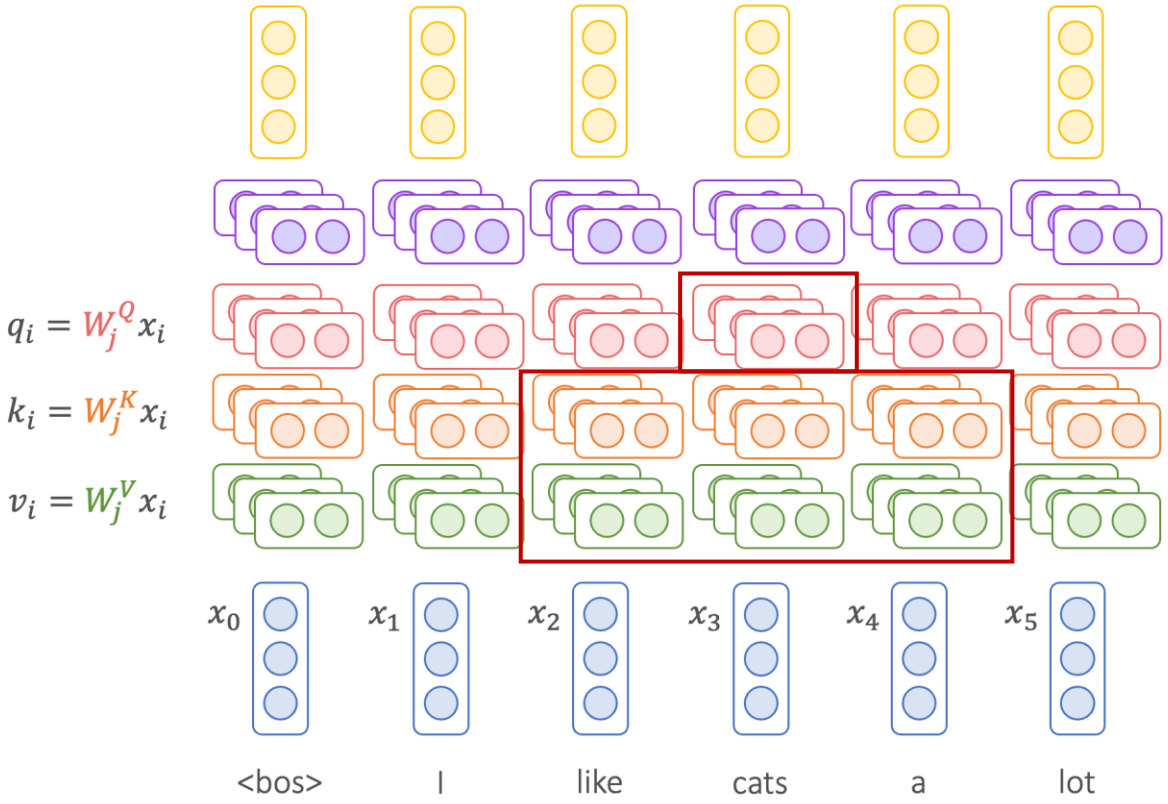
Sliding Window Attention Masking

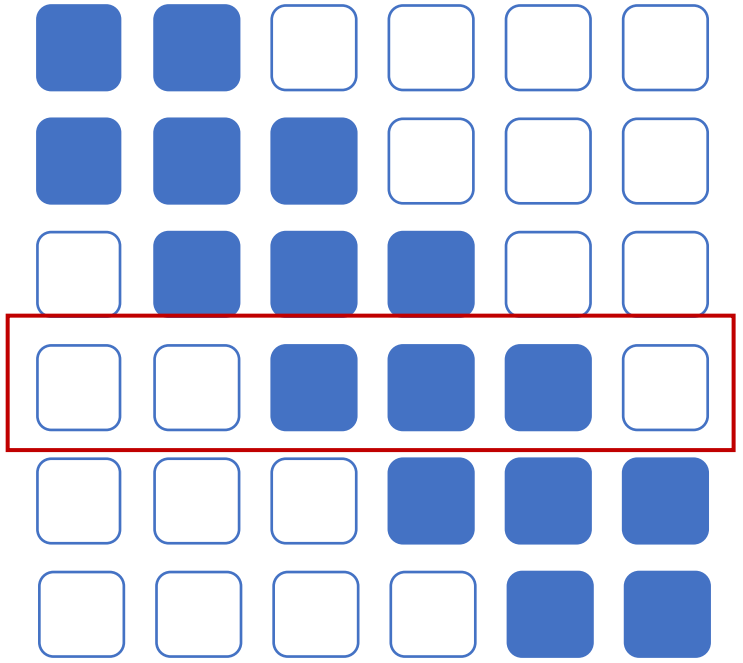




Sliding Window Attention Masking

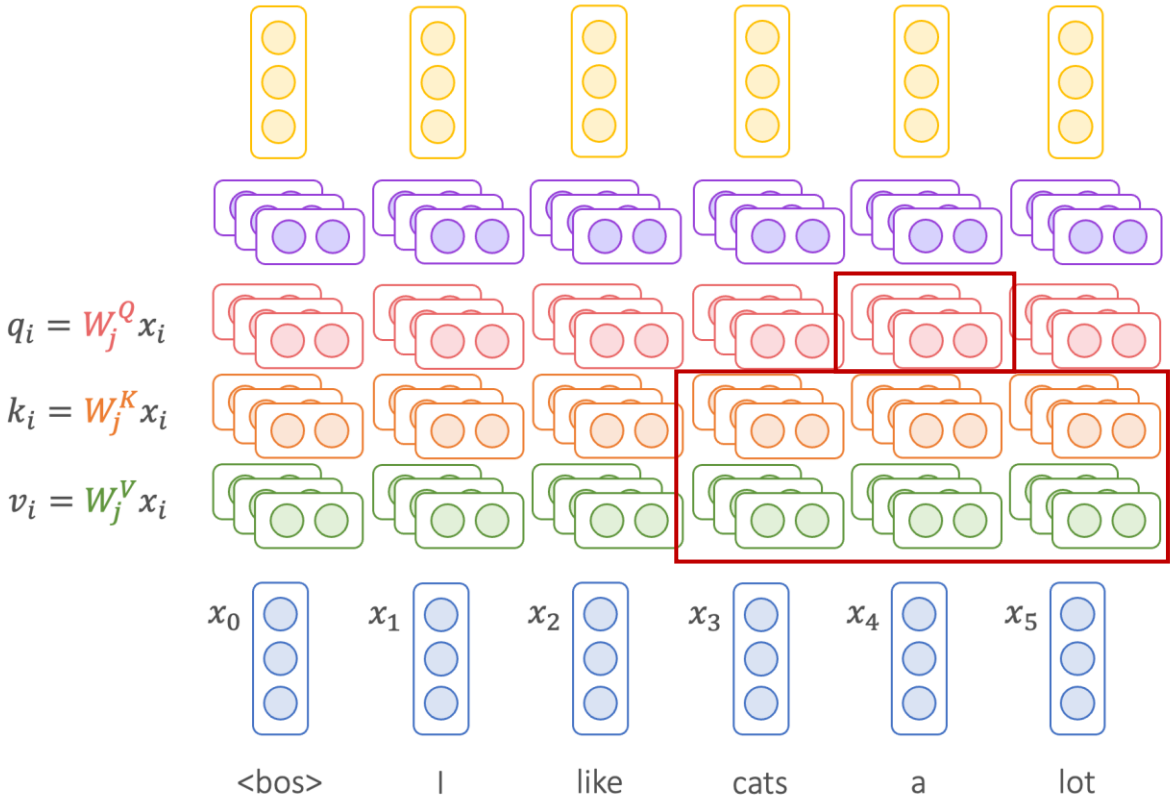
Sliding Window Attention Masking

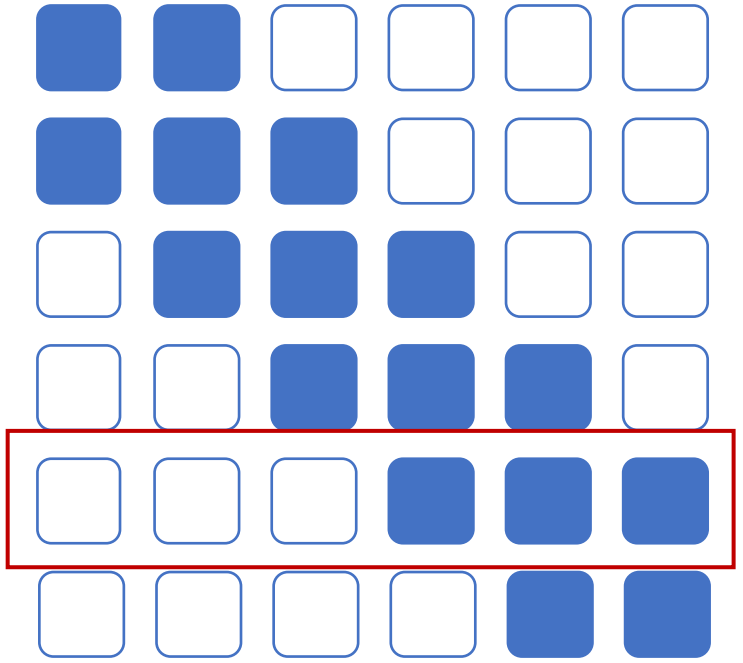




Sliding Window Attention Masking

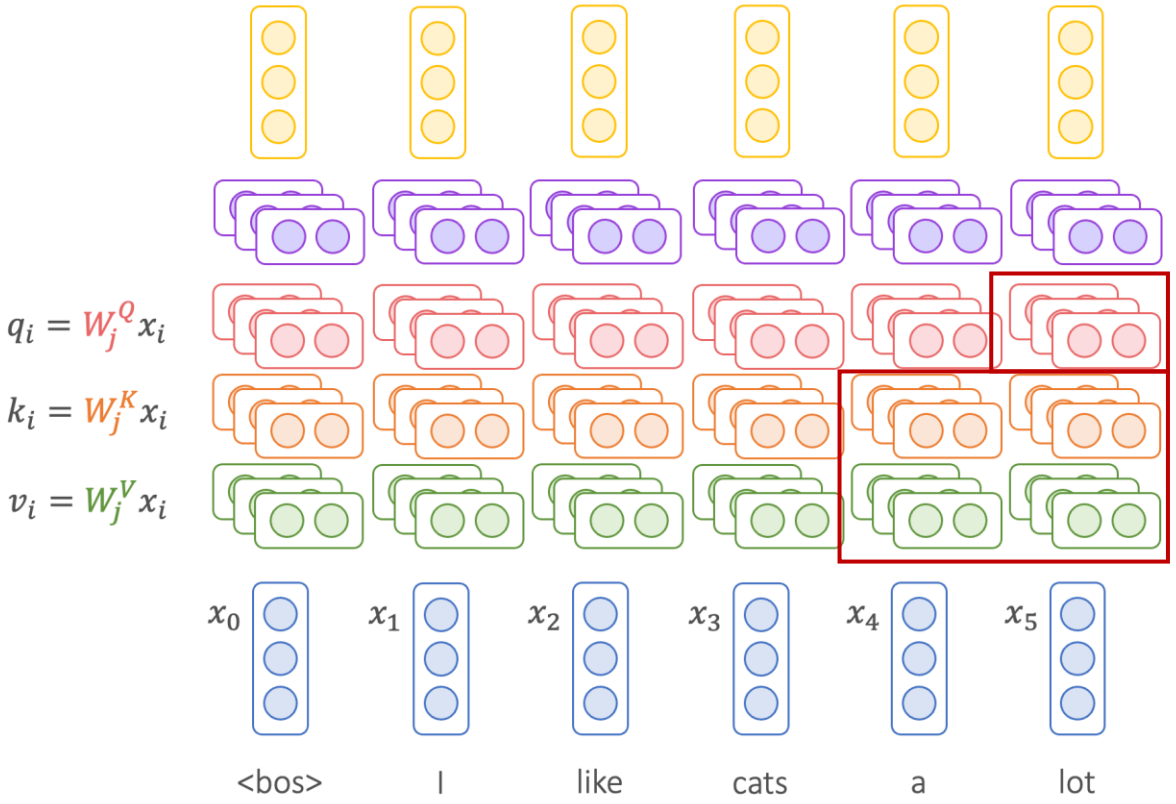
Sliding Window Attention Masking

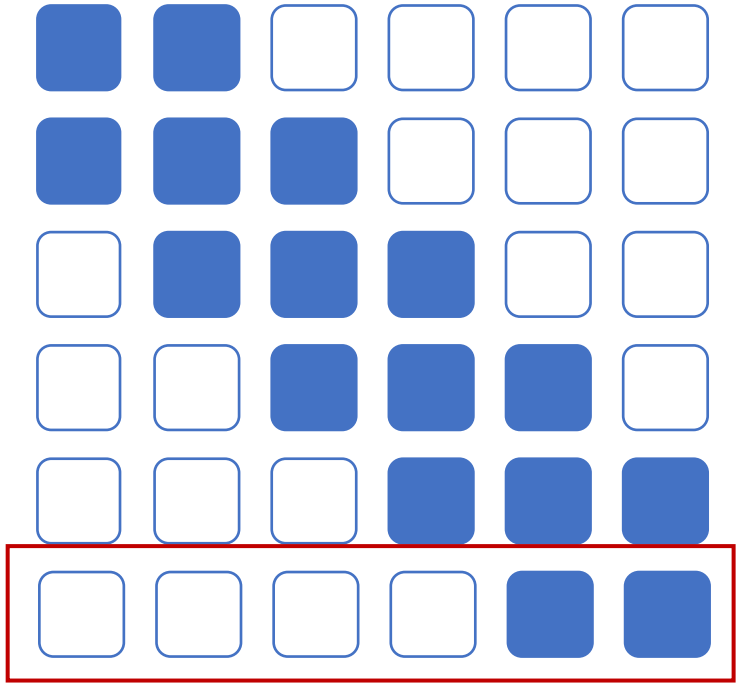




Sliding Window Attention Masking

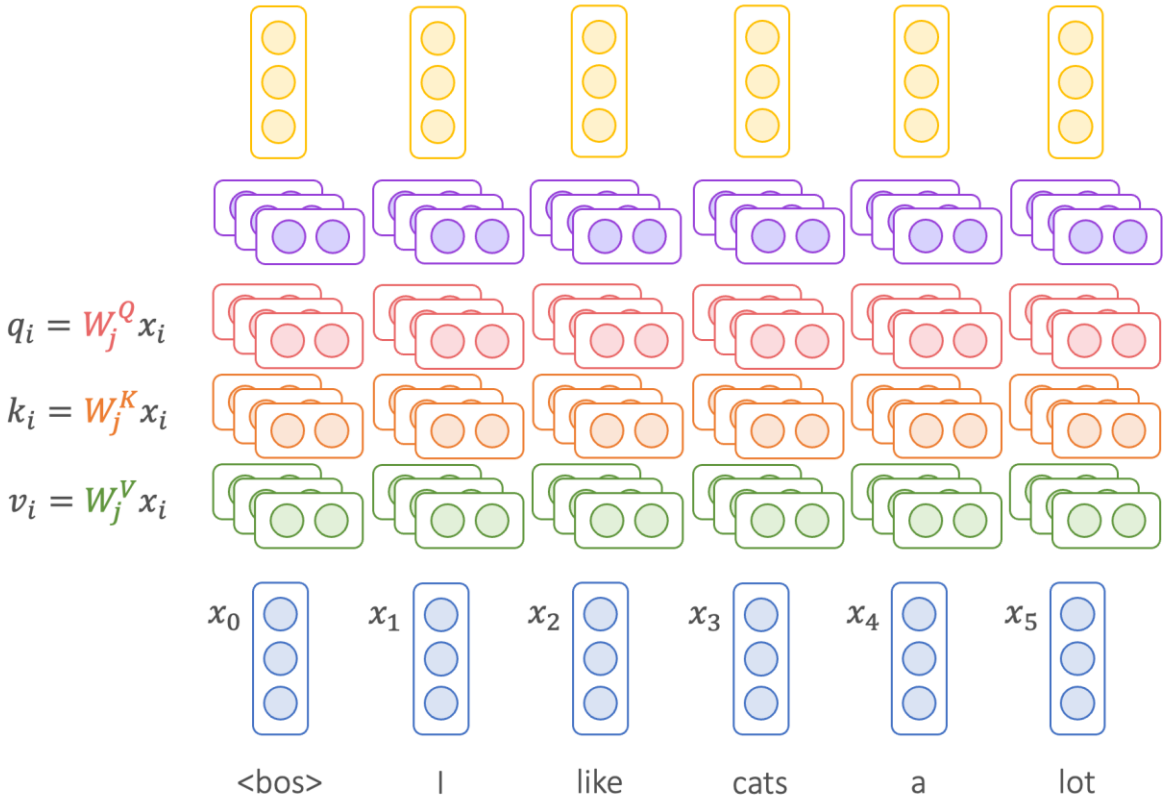
Sliding Window Attention Masking

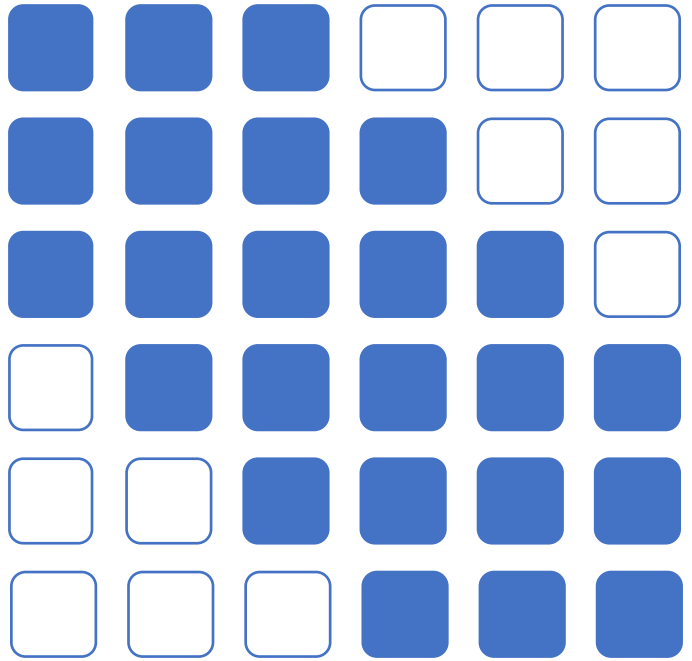




Sliding Window Attention Masking

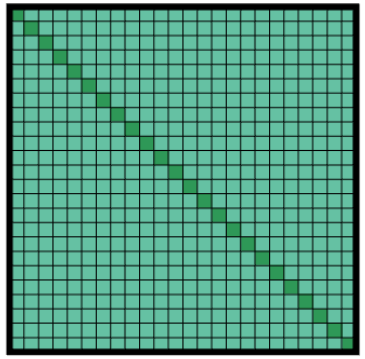
Sliding Window Attention Masking



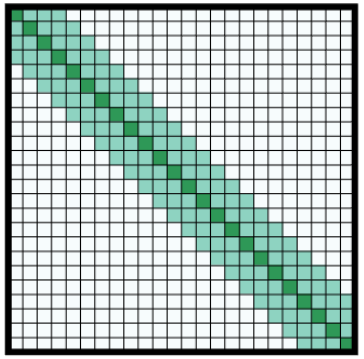


Sliding Window Attention Masking

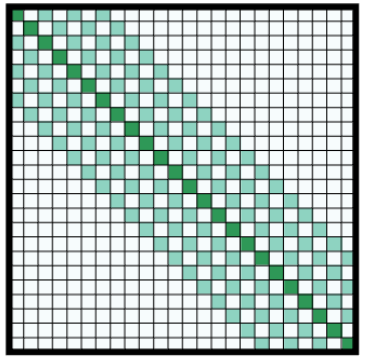
Different Types of Attention Masks



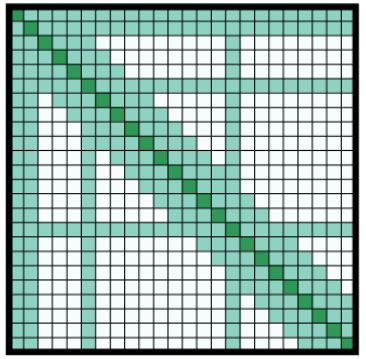
(a) Full n^2 attention



(b) Sliding window attention



(c) Dilated sliding window



(d) Global+sliding window

LongFormer Results on Language Modeling

Model	#Param	Dev	Test
Dataset `text8`			
T12 (Al-Rfou et al., 2018)	44M	-	1.18
Adaptive (Sukhbaatar et al., 2019)	38M	1.05	1.11
BP-Transformer (Ye et al., 2019)	39M	-	1.11
Our Longformer	41M	1.04	**1.10**
Dataset `enwik8`			
T12 (Al-Rfou et al., 2018)	44M	-	1.11
Transformer-XL (Dai et al., 2019)	41M	-	1.06
Reformer (Kitaev et al., 2020)	-	-	1.05
Adaptive (Sukhbaatar et al., 2019)	39M	1.04	1.02
BP-Transformer (Ye et al., 2019)	38M	-	1.02
Our Longformer	41M	1.02	**1.00**

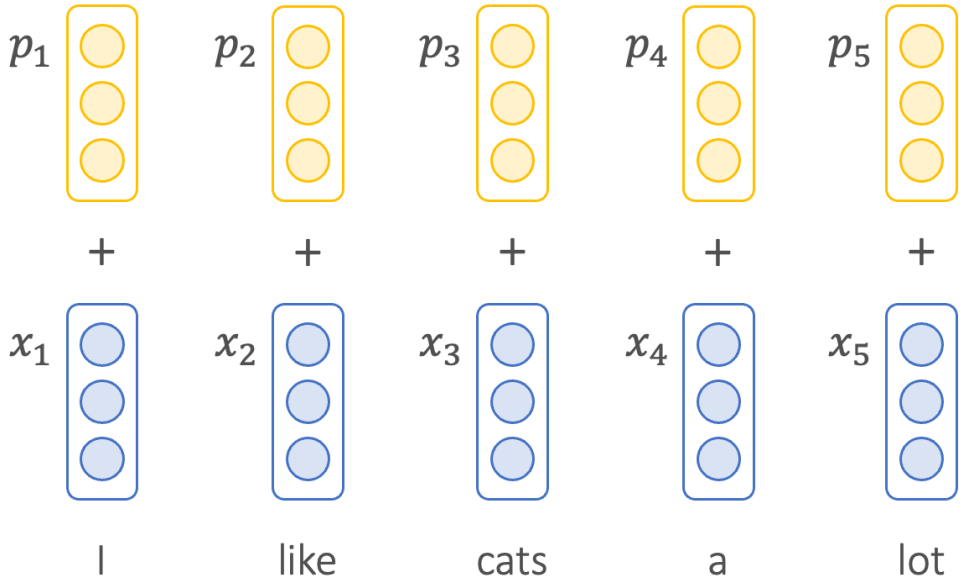
Longformer: The Long-Document Transformer, 2020

Lecture Plan

- Transformers
  - Encoder
  - Decoder
  - Encoder-Decoder
- Transformers Variants
  - Longformer
  - Relative Positional Encoding
  - RoFormer

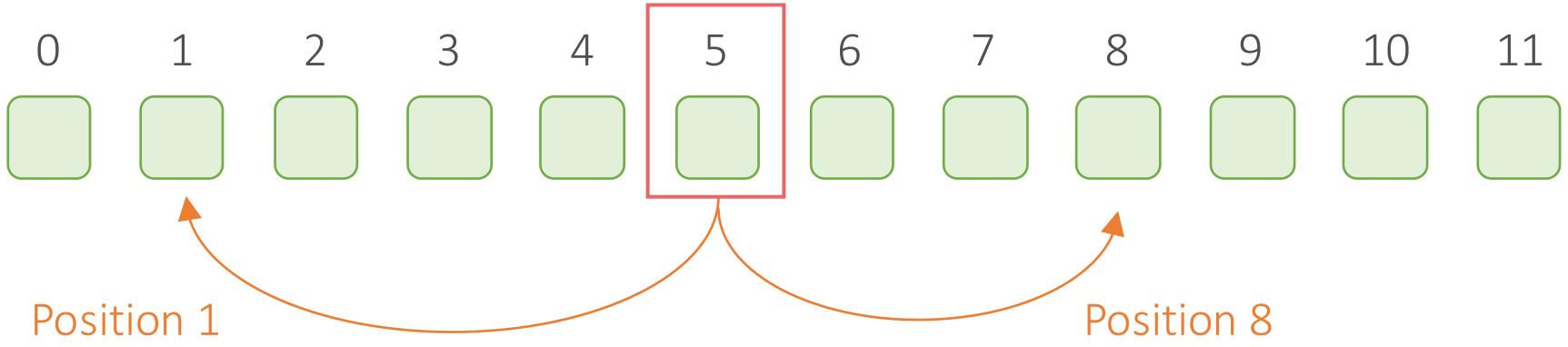
Absolute Positional Encoding

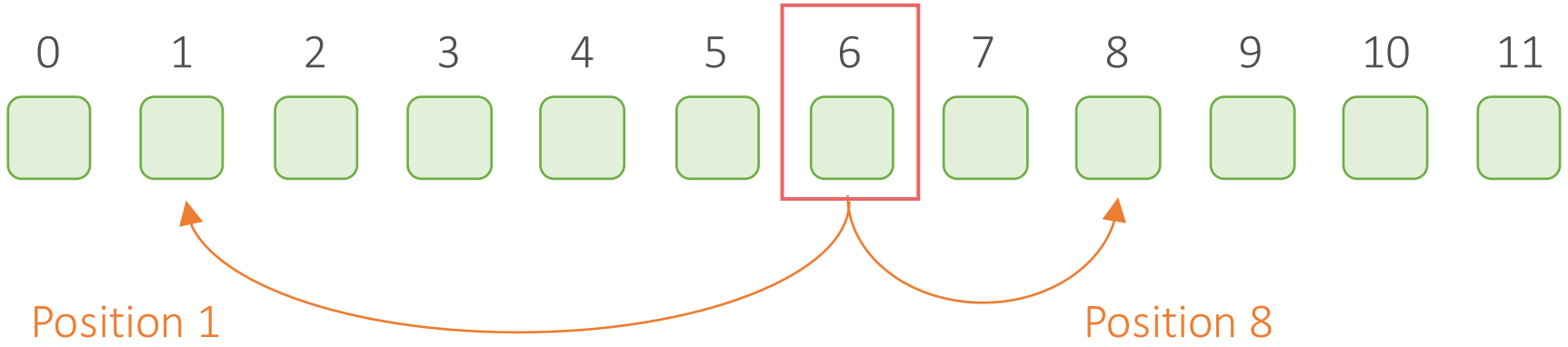
$$x_i \leftarrow x_i + PE_i$$



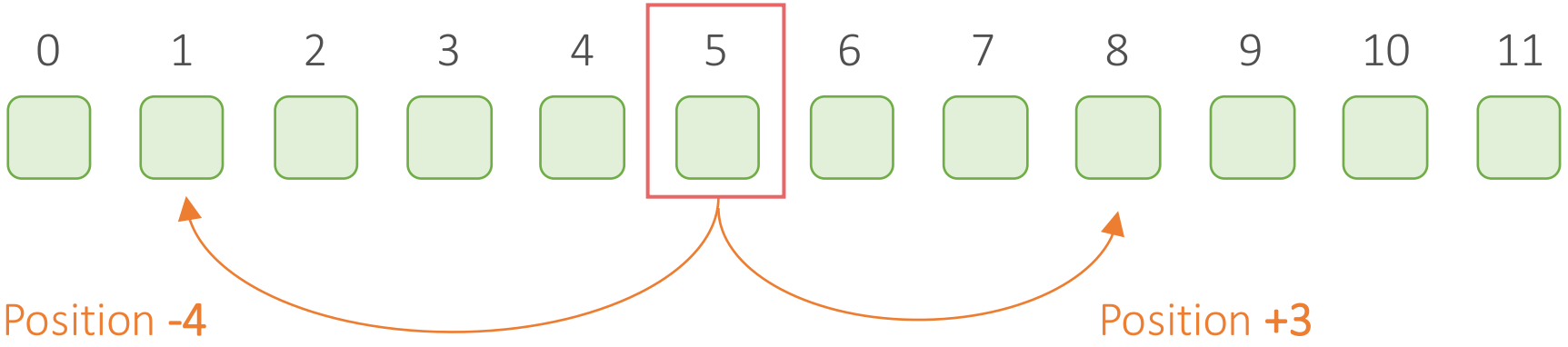
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

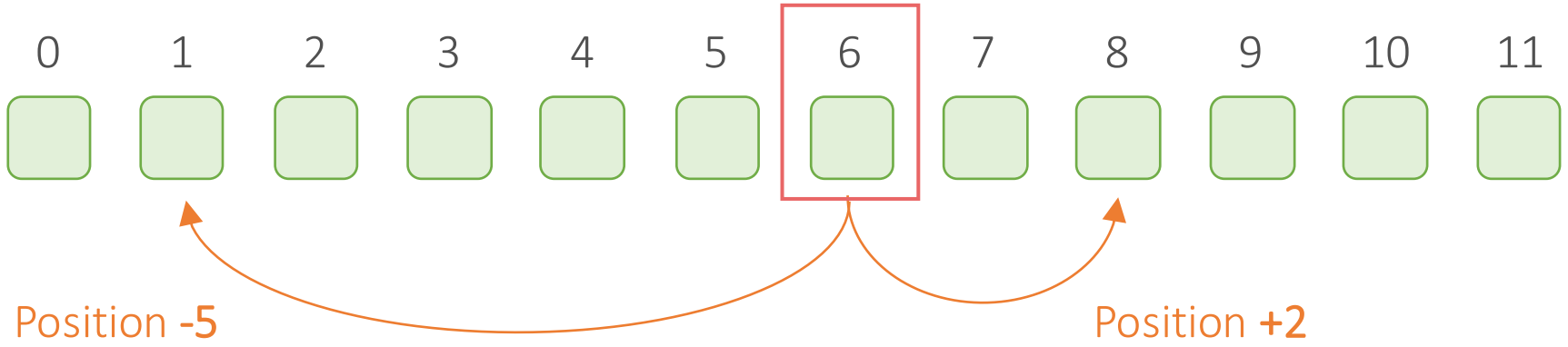
Absolute Position





Relative Position

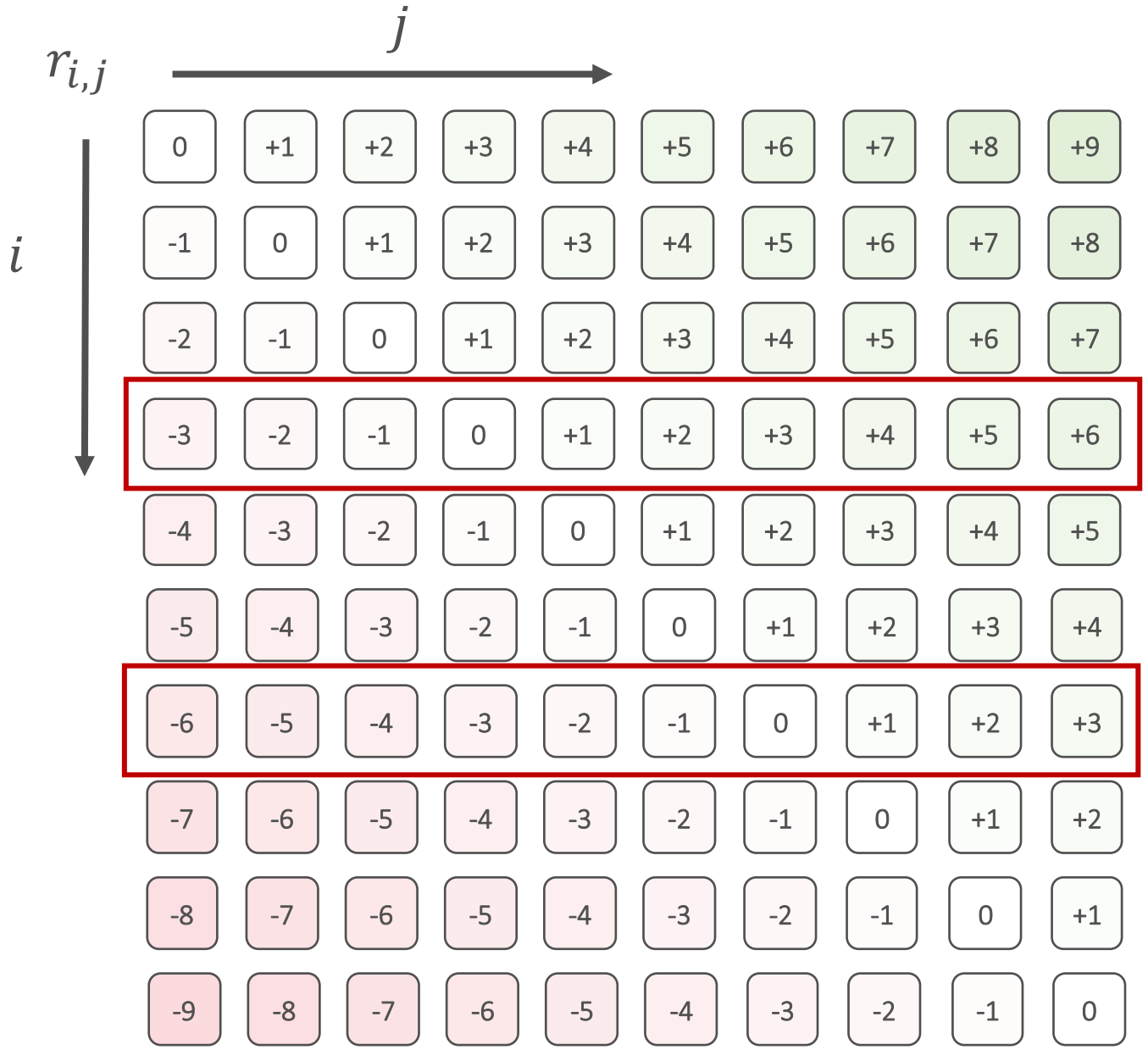




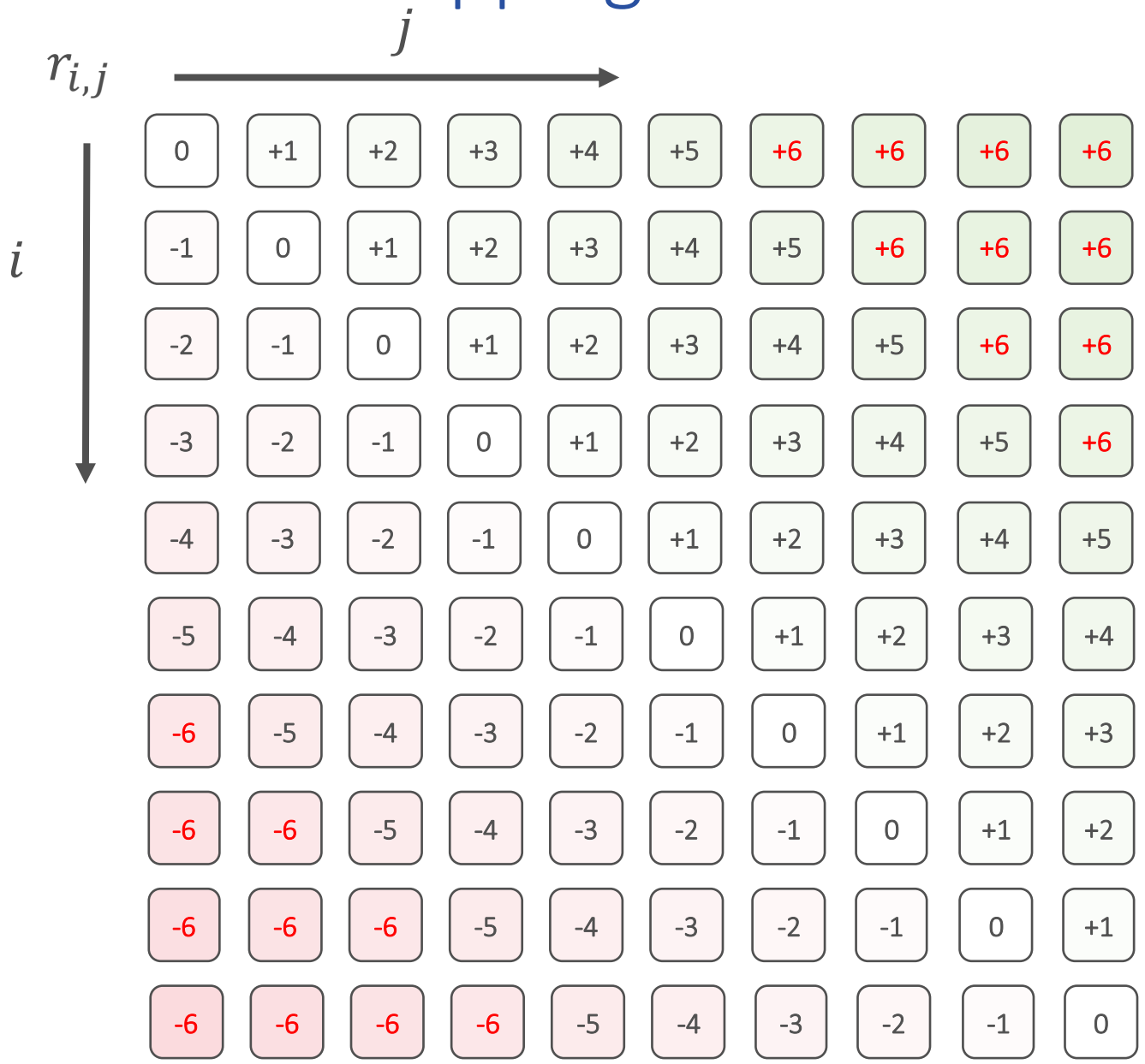
Why Relative Position?

- More contextual awareness
  - Position -4: 4 position before this word
  - Position +3: 4 position after this word
- Generalization to longer sequences

Relative Position

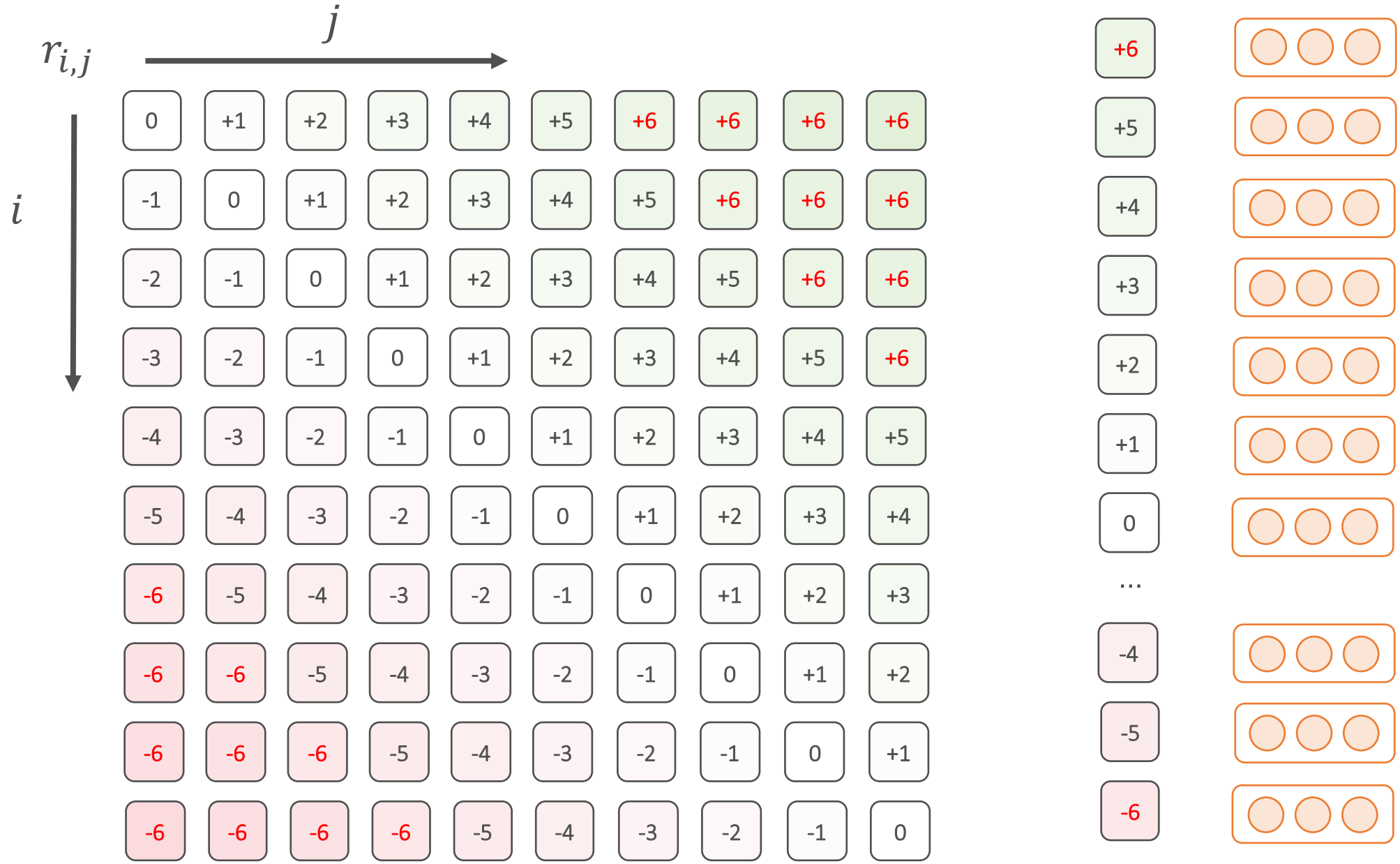


Relative Position with Clipping



Limited relative positions

Map Relative Positions to Embeddings



Self-Attention

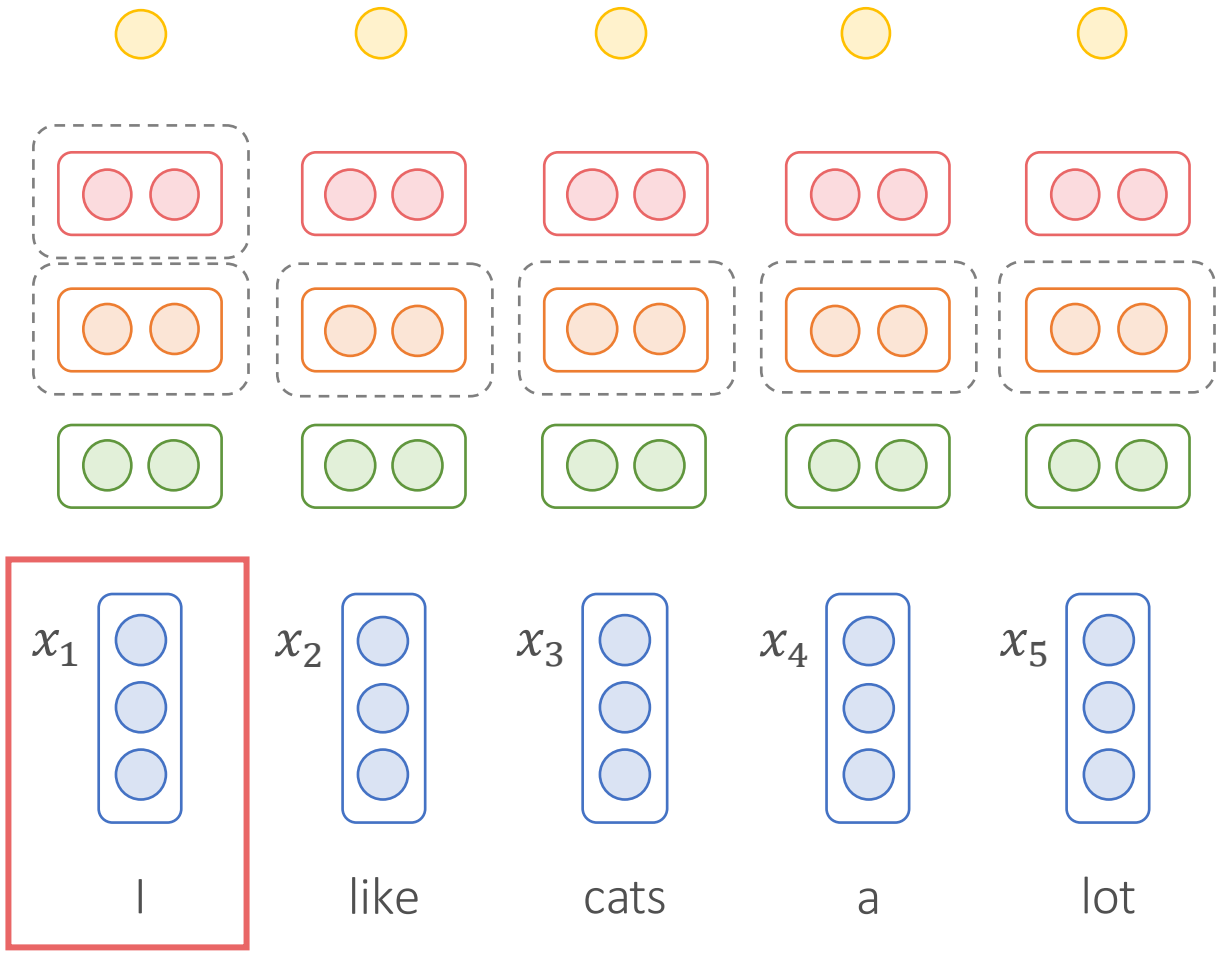
$$\alpha_{1,i} = \text{softmax}\left(\frac{q_1 \cdot k_i}{\sqrt{d}}\right)$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention

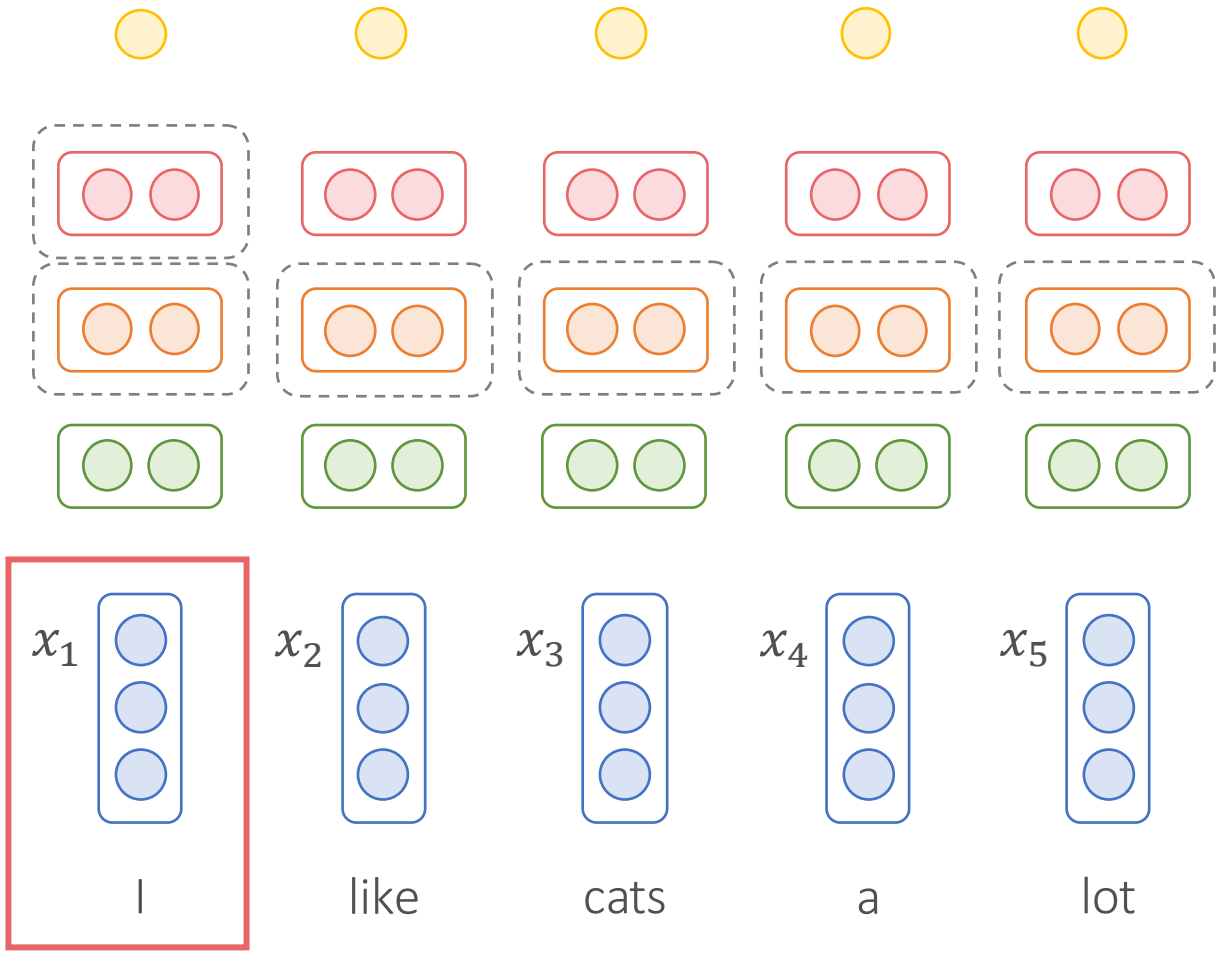
$$\alpha_{1,i} = \text{softmax}\left(\frac{W^Q x_1 \cdot W^K x_i}{\sqrt{d}}\right)$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention with Relative Position Embeddings

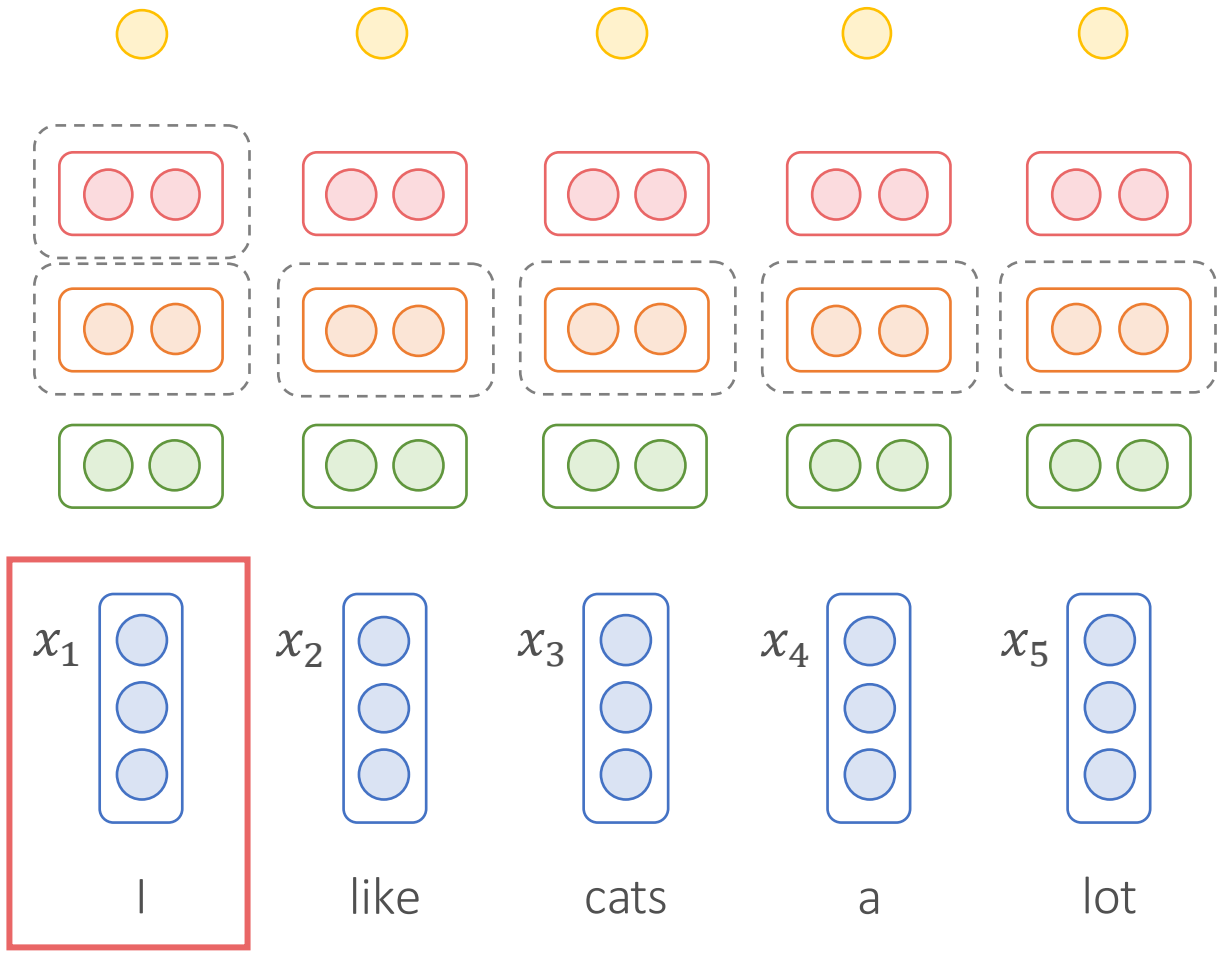
$$\alpha_{1,i} = \text{softmax}\left(\frac{W^Q x_1 \cdot W^K\left(x_i + RE(r_{1,i})\right)}{\sqrt{d}}\right)$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention with Relative Position Embeddings

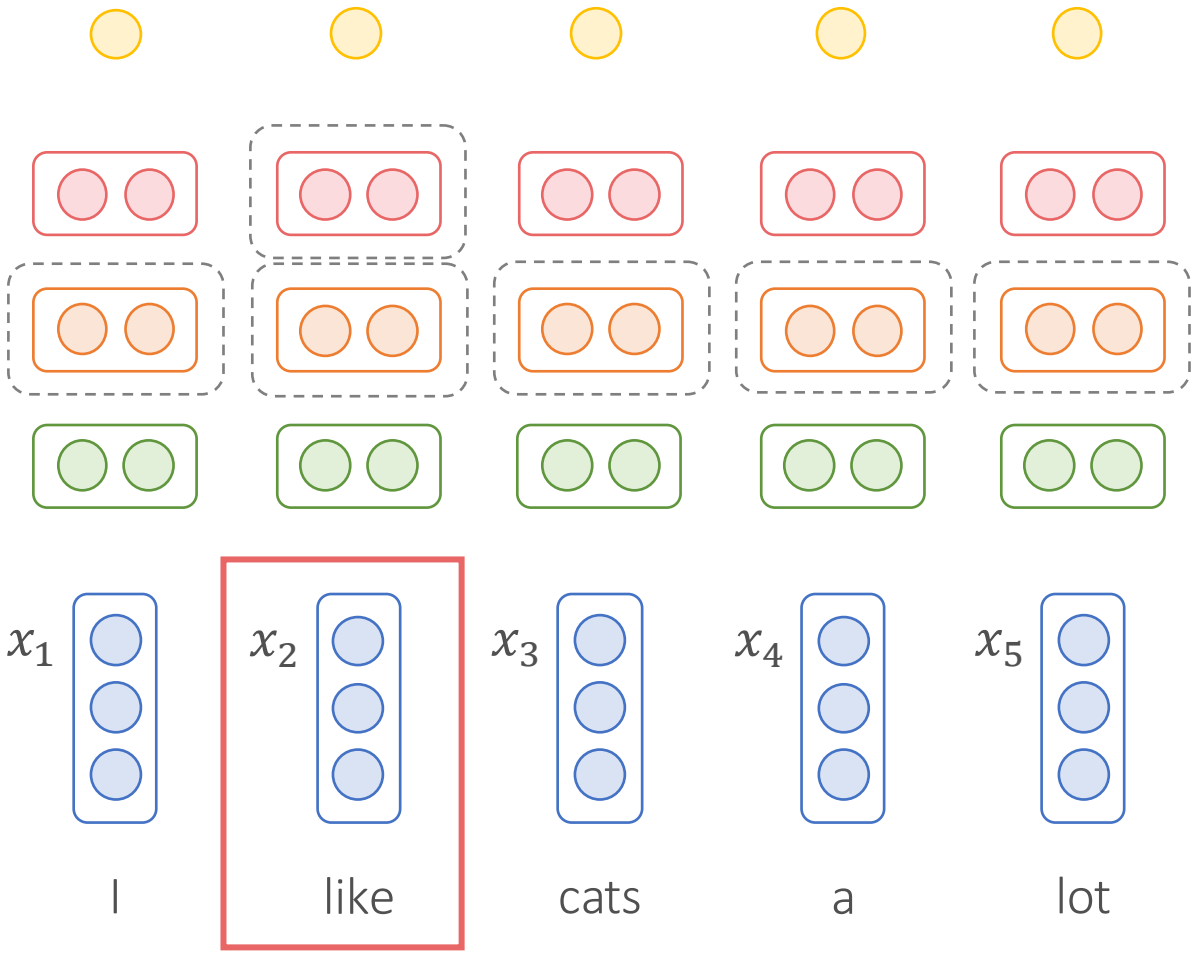
$$\alpha_{2,i} = \text{softmax}\left(\frac{W^Q x_2 \cdot W^K\left(x_i + RE(r_{2,i})\right)}{\sqrt{d}}\right)$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Relative Positions for Machine Translation

Model	Position Information	EN-DE BLEU	EN-FR BLEU
Transformer (base)	Absolute Position Representations	26.5	38.2
Transformer (base)	Relative Position Representations	**26.8**	**38.7**
Transformer (big)	Absolute Position Representations	27.9	41.2
Transformer (big)	Relative Position Representations	**29.2**	**41.5**

RoFormer

- Improved version of relative positional encoding
  - Rotary Position Embedding (RoPE)
- Most advanced large language models use RoPE

Self-Attention with Relative Position Embeddings

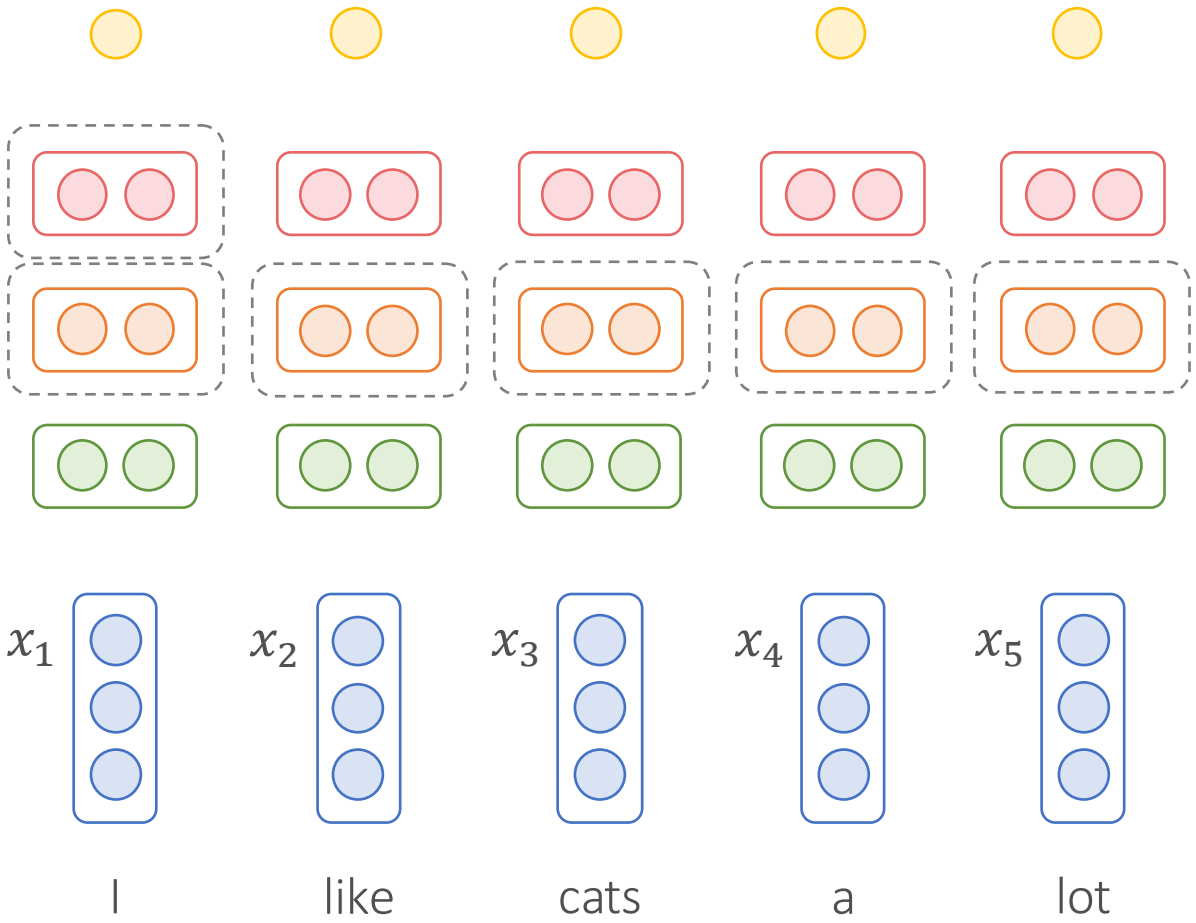
$$\alpha_{m,n} = \text{softmax}\left(\frac{W^Q x_m \cdot W^K\left(x_n + RE(r_{m,n})\right)}{\sqrt{d}}\right)$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention with RoPE (In 2D Case)

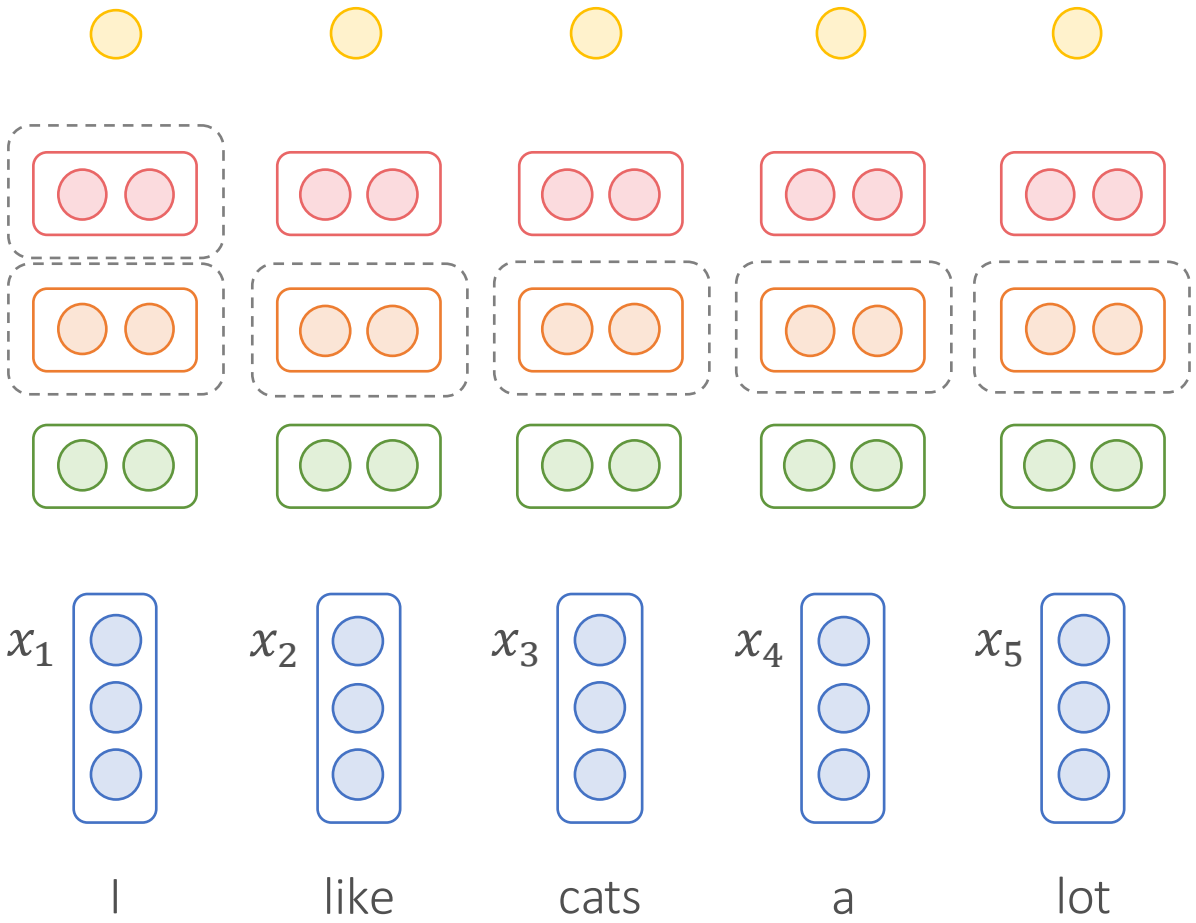
$$\alpha_{m,n} = \text{softmax}\left(\frac{\langle (W^Q x_m) e^{im\theta} \cdot (W^K x_n) e^{in\theta} \rangle}{\sqrt{d}}\right)$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention with RoPE (In 2D Case)

Equivalent to rotate $W^Q x_m$ with angle $m\theta$

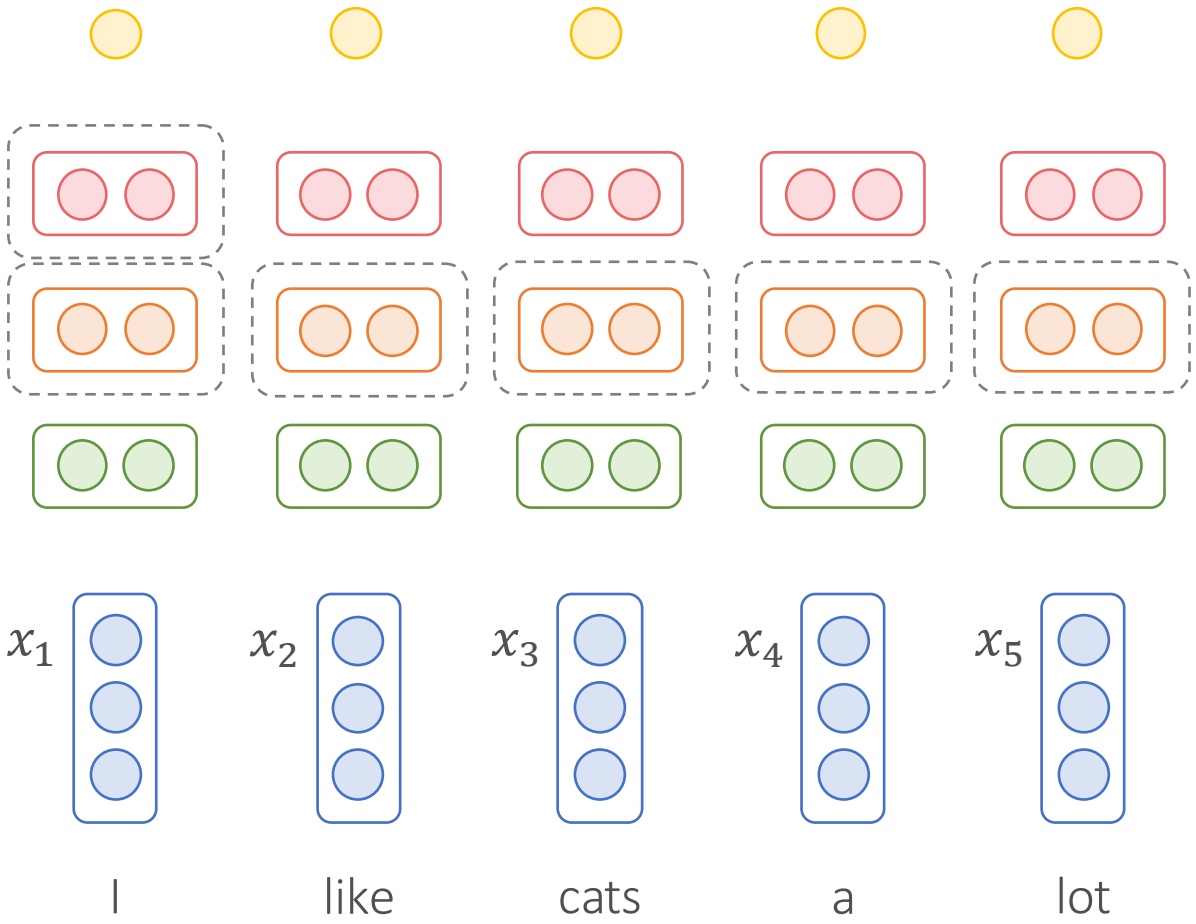
$$\alpha_{m,n} = \text{softmax}\left(\frac{\langle (W^Q x_m) e^{im\theta} \cdot (W^K x_n) e^{in\theta} \rangle}{\sqrt{d}}\right)$$

Normalized Attention Scores

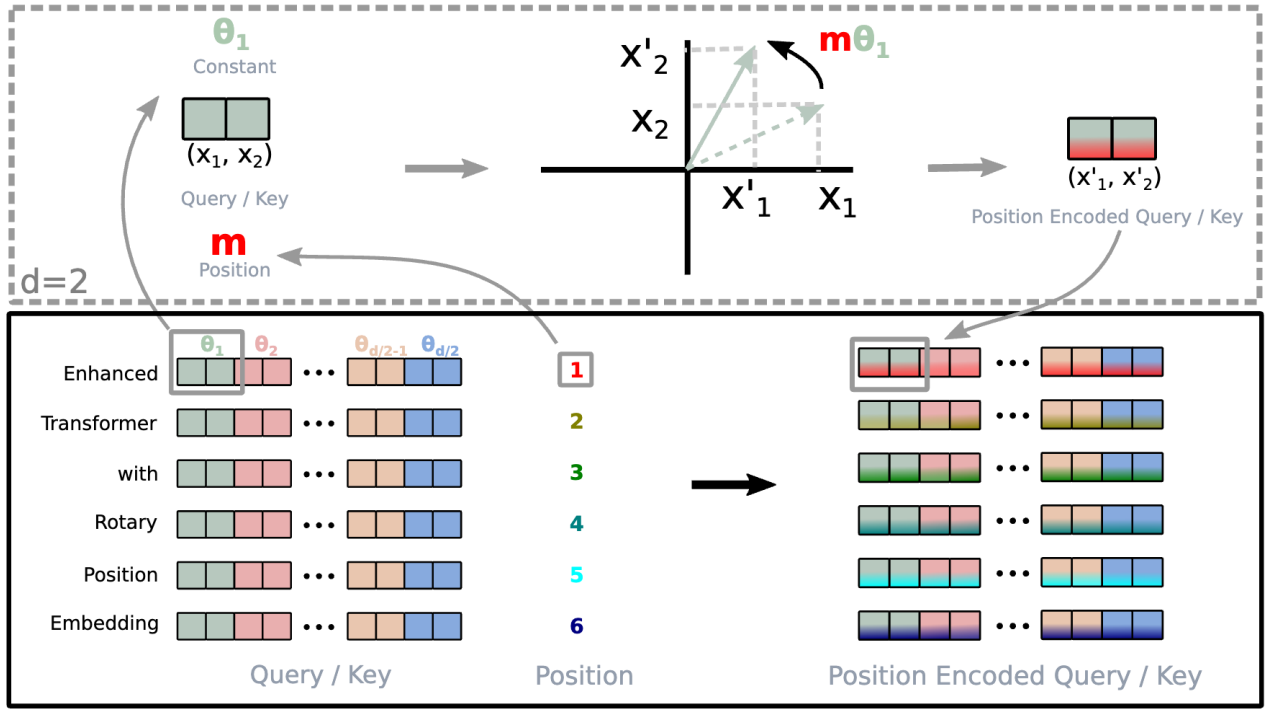
Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention with RoPE (In 2D Case)



$$\alpha_{m,n} = \text{softmax}\left(\frac{\langle (W^Q x_m) e^{im\theta} \cdot (W^K x_n) e^{in\theta} \rangle}{\sqrt{d}}\right)$$

$$f_q(\boldsymbol{x}_m, m) = (\boldsymbol{W}_q \boldsymbol{x}_m) e^{im\theta}$$

$$f_k(\boldsymbol{x}_n, n) = (\boldsymbol{W}_k \boldsymbol{x}_n) e^{in\theta}$$

$$\langle f_q(\boldsymbol{x}_m, m), f_k(\boldsymbol{x}_n, n) \rangle = \text{Re}[(\boldsymbol{W}_q \boldsymbol{x}_m)(\boldsymbol{W}_k \boldsymbol{x}_n)^* e^{i(m-n)\theta}]$$

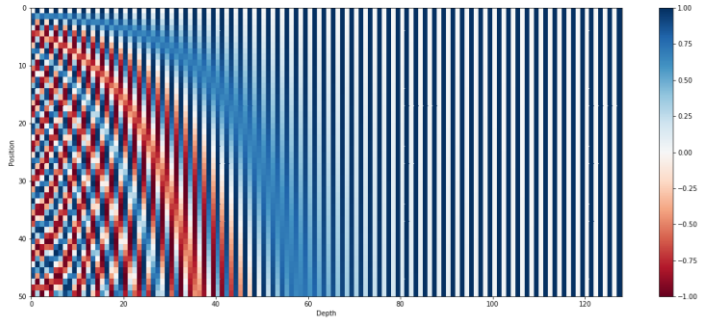
General Form of RoPE

$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_{\{q,k\}}\boldsymbol{x}_m$$

Different base angle $\theta_1, \theta_2, \ldots, \theta_{d/2}$

$$\boldsymbol{R}^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = (\boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_q\boldsymbol{x}_m)^\intercal(\boldsymbol{R}^d_{\Theta,n}\boldsymbol{W}_k\boldsymbol{x}_n) = \boldsymbol{x}^\intercal\boldsymbol{W}_q R^d_{\Theta,n-m}\boldsymbol{W}_k\boldsymbol{x}_n$$



Similar to the idea of using different flipping frequency for Sinusoidal positional encoding

RoPE Performance

Model	MRPC	SST-2	QNLI	STS-B	QQP	MNLI(m/mm)
BERTDevlin et al. [2019]	88.9	93.5	90.5	85.8	71.2	84.6/83.4
RoFormer	**89.5**	90.7	88.0	**87.0**	**86.4**	80.2/79.8