CSCE 689: Special Topics in Advanced NLP

Lecture 5: Transformers, Pre-Training

Kuan-Hao Huang

Fall 2026

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

LaTeX Assignment

- LaTeX Assignment (1%) [Due: 9/10]
- No late submission

CSCE 689: LaTeX Assignment

Your Name
Your UID and email

Overview

This assignment is designed to give you practice with LaTeX, which you are expected to use for your literature review, project proposal, and final report in this course.

Instructions

For this assignment, you will create a PDF containing your answers using LaTeX. If this is your first time working with LaTeX, we recommend starting with this short tutorial, which covers the basic features you will need for this course. Please use the Association for Computational Linguistics LaTeX template (link), a template widely used in major NLP conferences. We suggest using Overleaf as your online editor, since it automatically manages packages for you.

By default, the template is set to *review mode*. To switch to *final mode*, change:

Review Mode (Default)

```
\usepackage[review]{acl}
```

to:

Final Mode

```
\usepackage[final]{acl}
```

This allows you to display the author information. Be sure to include your name, UIN, and email.

The following sections contain questions on some commonly used LaTeX commands. There are a total of 100 points for this assignment. Please answer each question in a separate *section*, and submit the final PDF generated using LaTeX.

1 Including Equations [20pts]

Typeset the following expression using LaTeX:

$$\frac{\partial \mathcal{L}_{\text{total}}}{\partial \mathbf{w}_j} = -\frac{1}{m}\sum_{i=1}^{m}\left(y_i - \sigma(z_i)\right)\cdot \mathbf{x}_{i,j}$$

2 Including Images [20pts]

Select a picture of a cat and include it with a caption. The figure below is provided as an example.

Figure 1: This is a cute cat!

3 Including Tables [20pts]

Create a table that displays your name, UIN, and email. You can follow the example below as a template.

Name	Kuan-Hao Huang
UIN	123456789
Email	khhuang@tamu.edu

Table 1: Example table.

4 Including Lists [20pts]

Create a list that displays your name, UIN, and email. You can follow the example below as a template.

- Name: Kuan-Hao Huang
- UIN: 123456789
- Email: khhuang@tamu.edu

5 Including Citations [20pts]

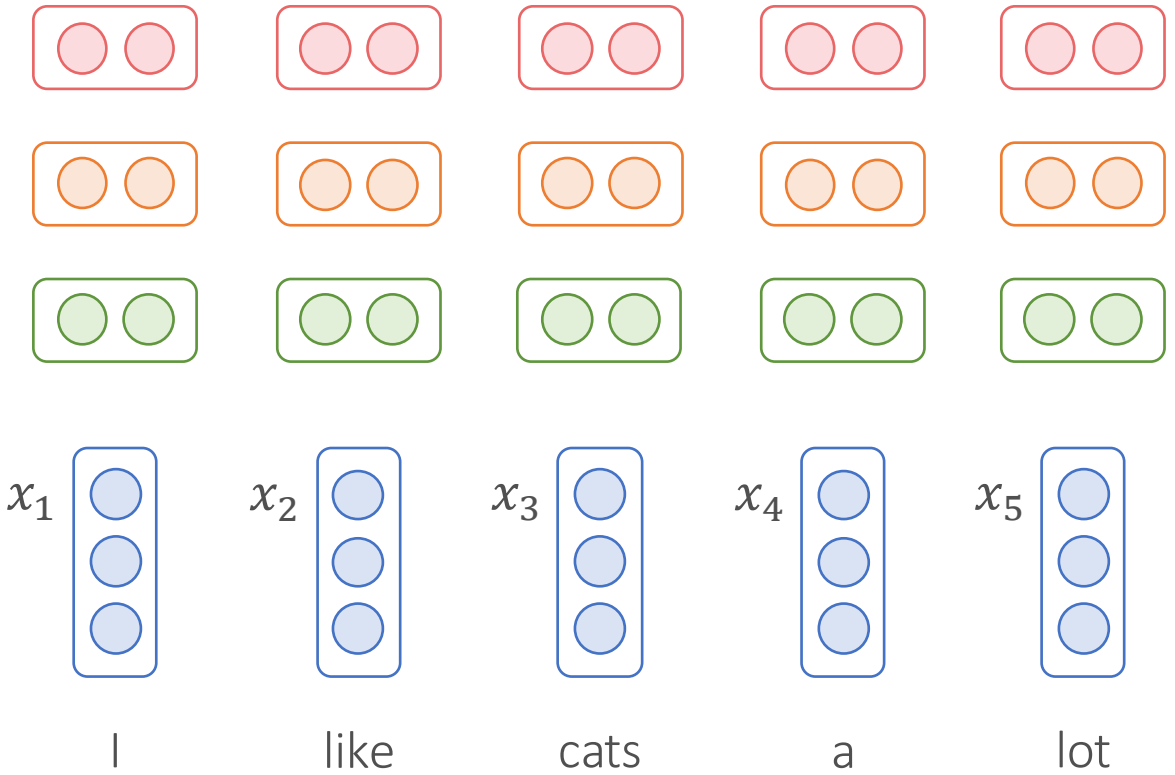
Use *BibTex* to include the following paper: Paper 1 (Vaswani et al., 2017) and Paper 2 (Devlin et al., 2019). You can learn more about *BibTex* here.

Self-Attention

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



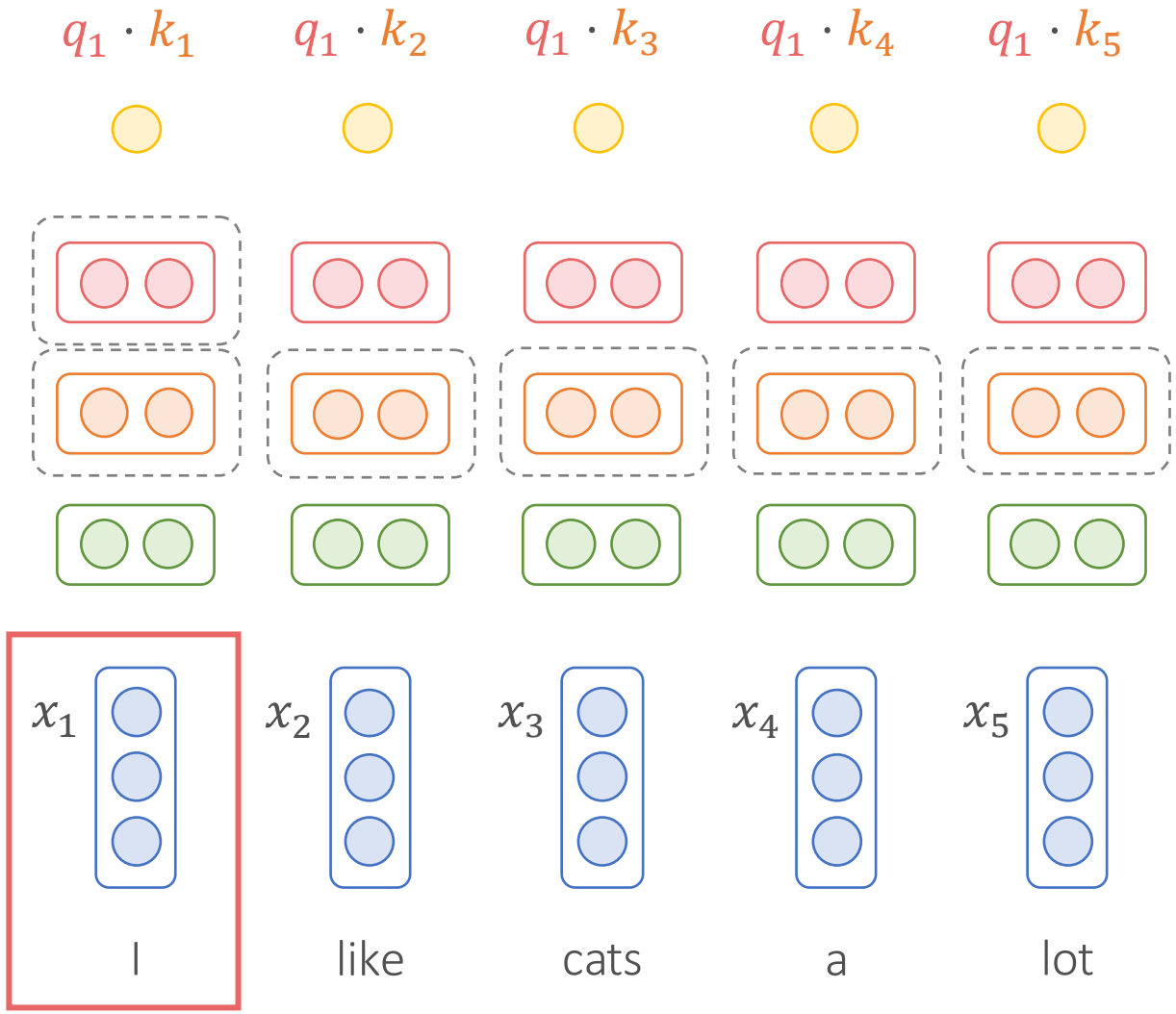
Self-Attention

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

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

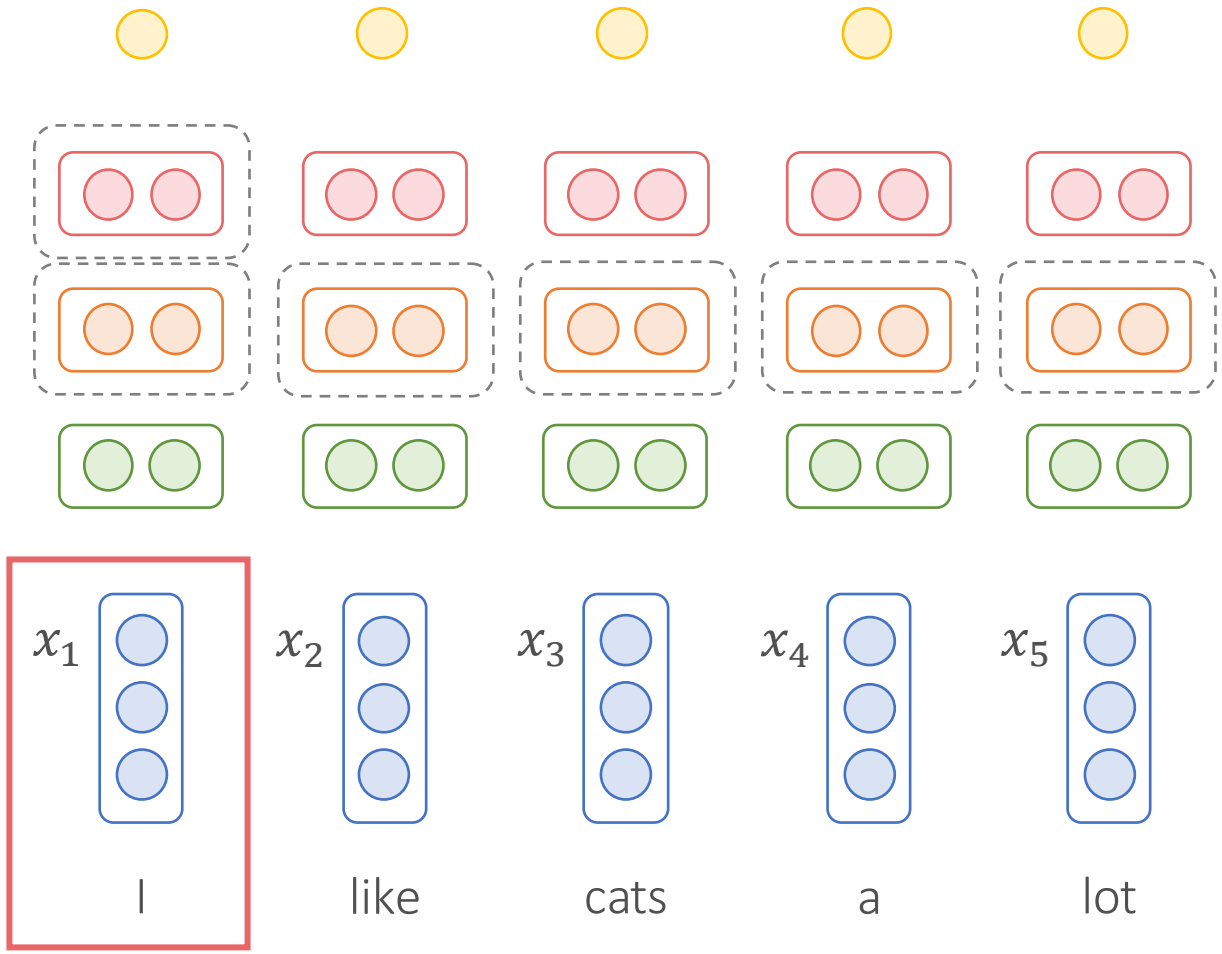
Vector dimension

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention

Weighted Sum

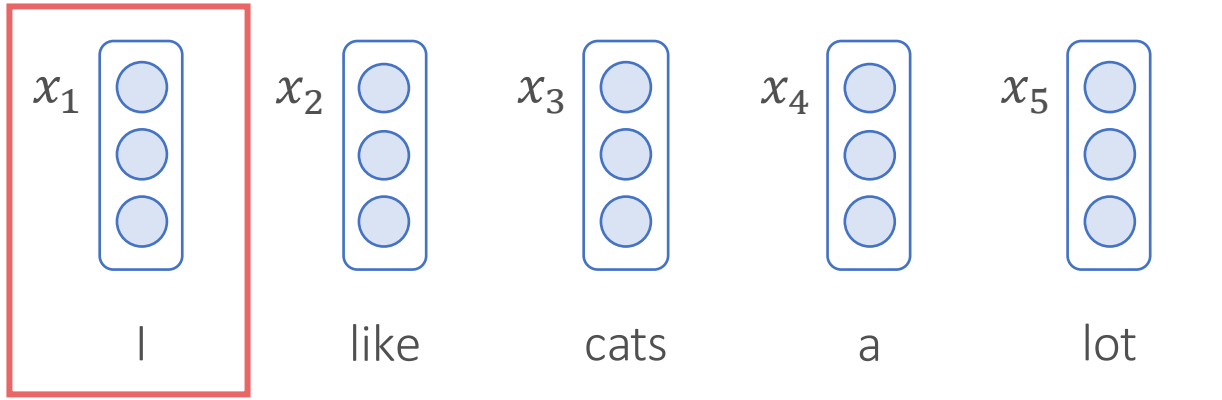
$$z_1 = \sum_i \alpha_{1,i} v_i$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



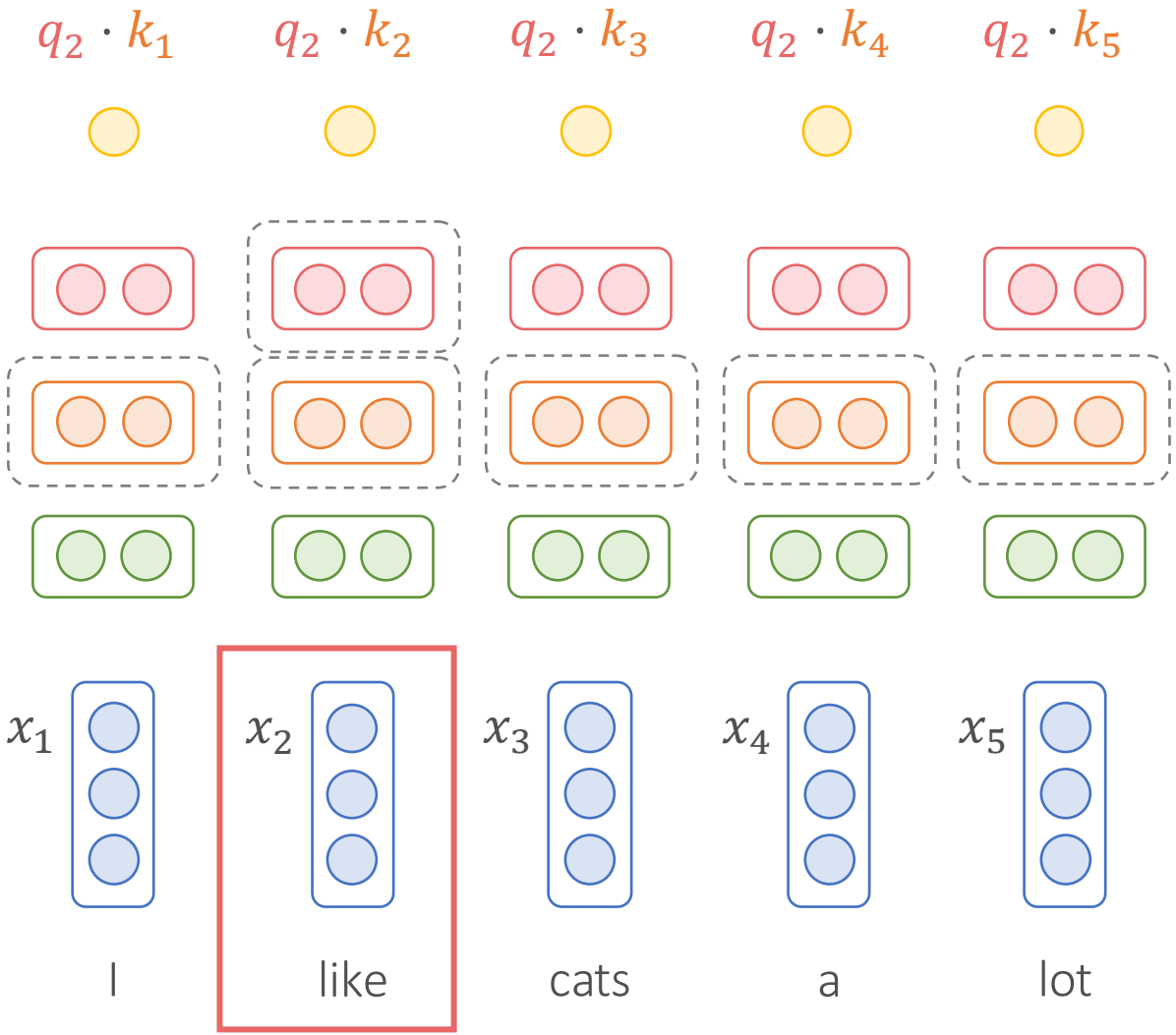
Self-Attention

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

$$\alpha_{2,i} = \text{softmax}\left(\frac{q_2 \cdot k_i}{\sqrt{d}}\right)$$

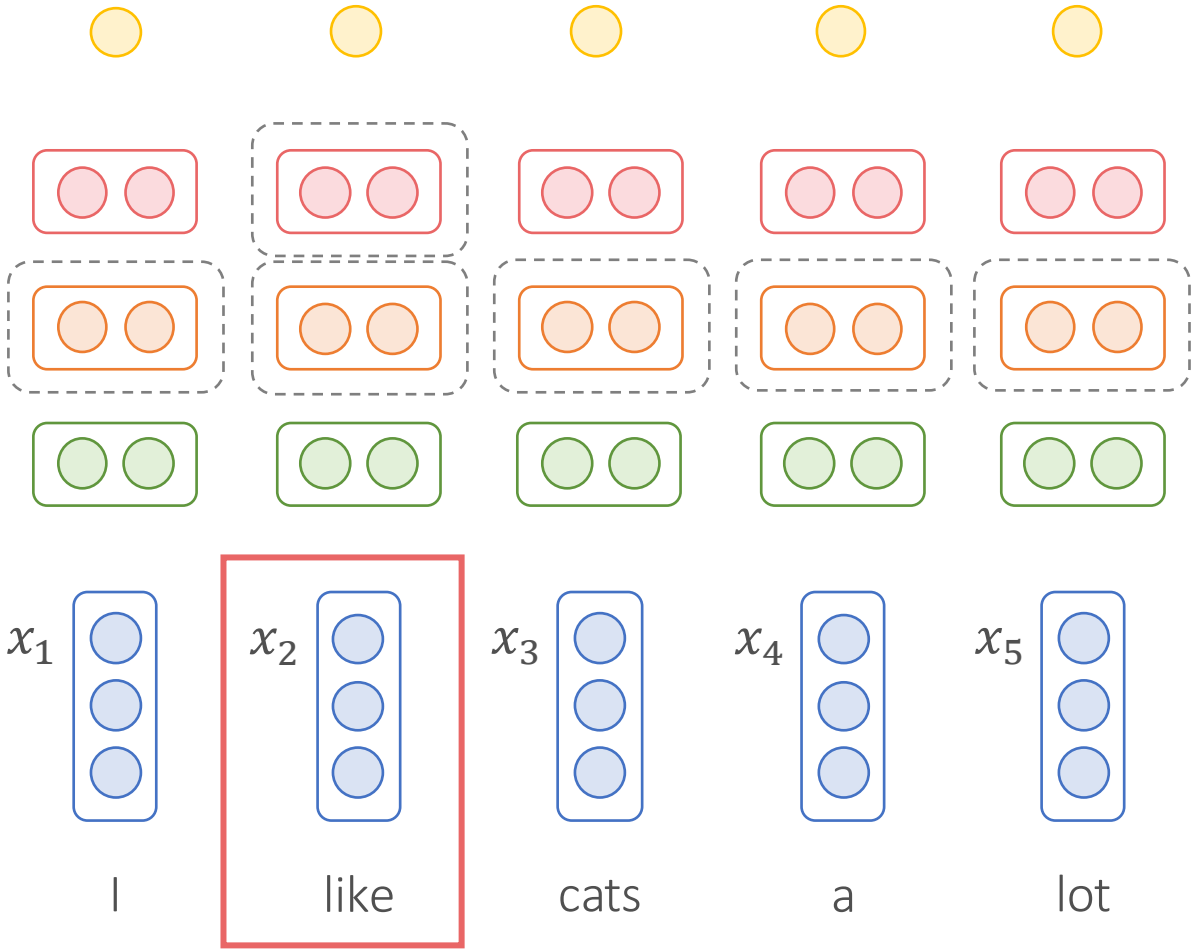
Vector dimension

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention

Weighted Sum

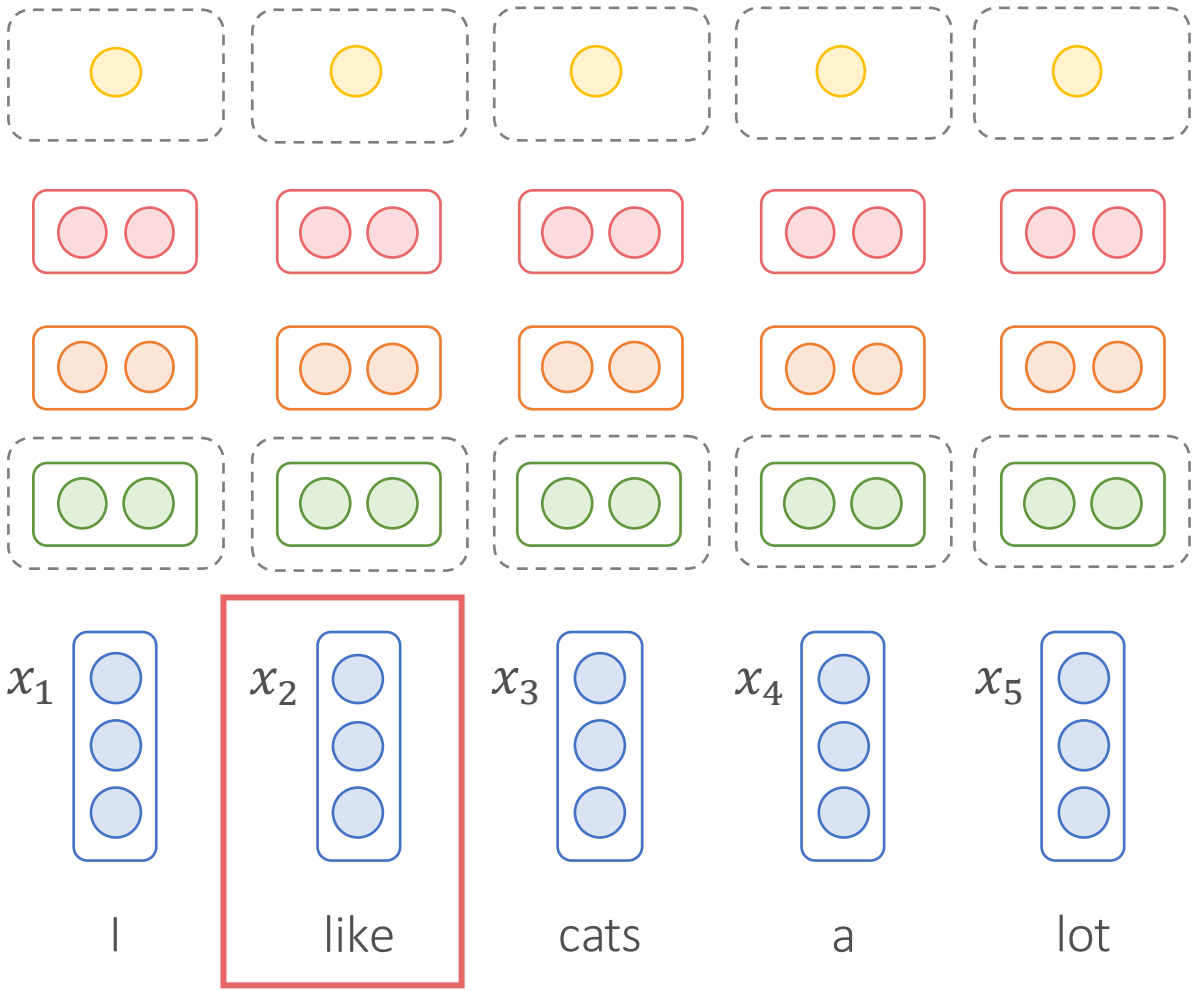
$$z_2 = \sum_i \alpha_{2,i} v_i$$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



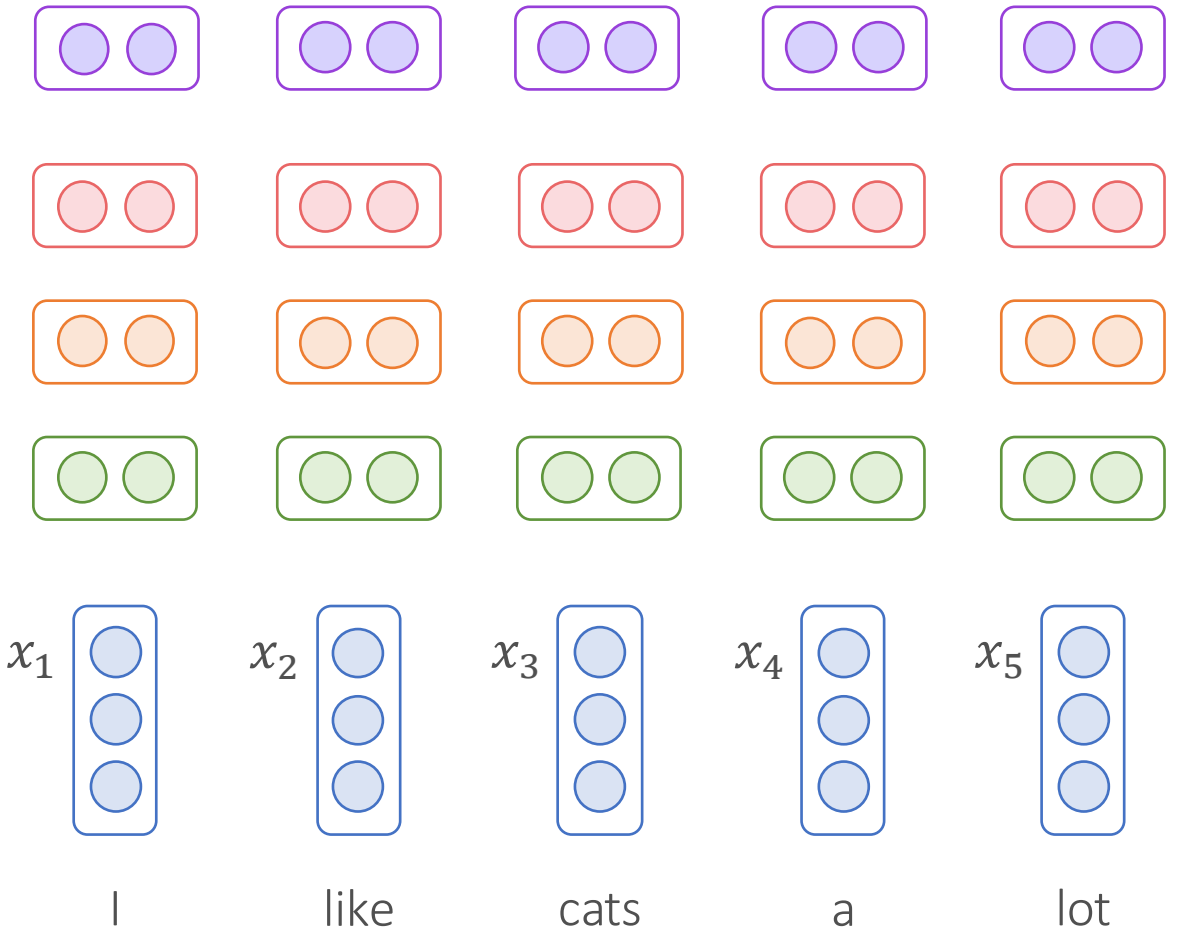
Self-Attention

Self-Attention Output

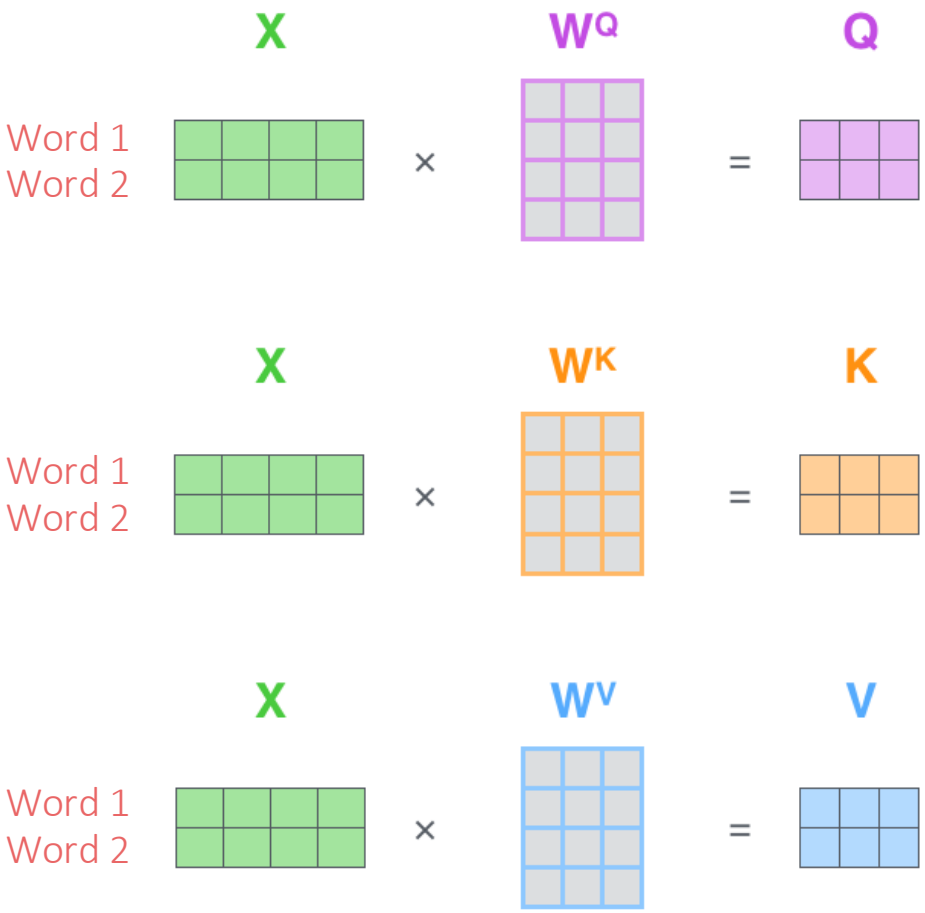
Query $q_i = W^Q x_i$

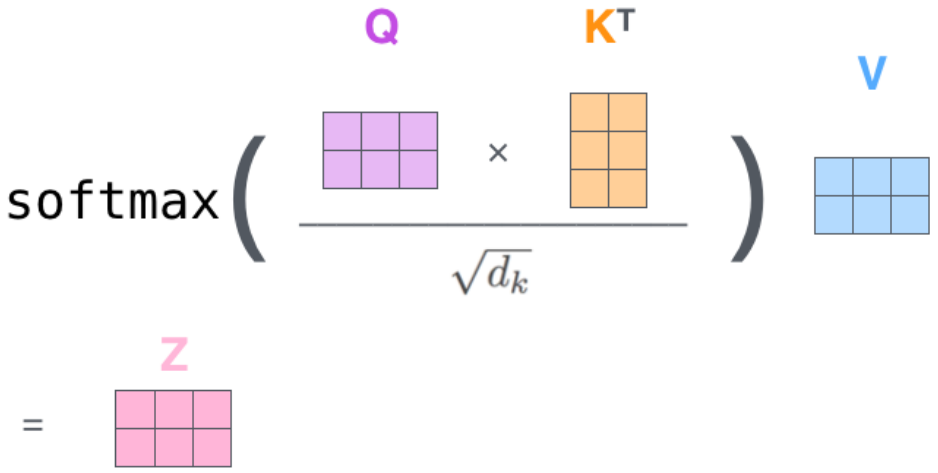
Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention – Matrix Form





$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

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

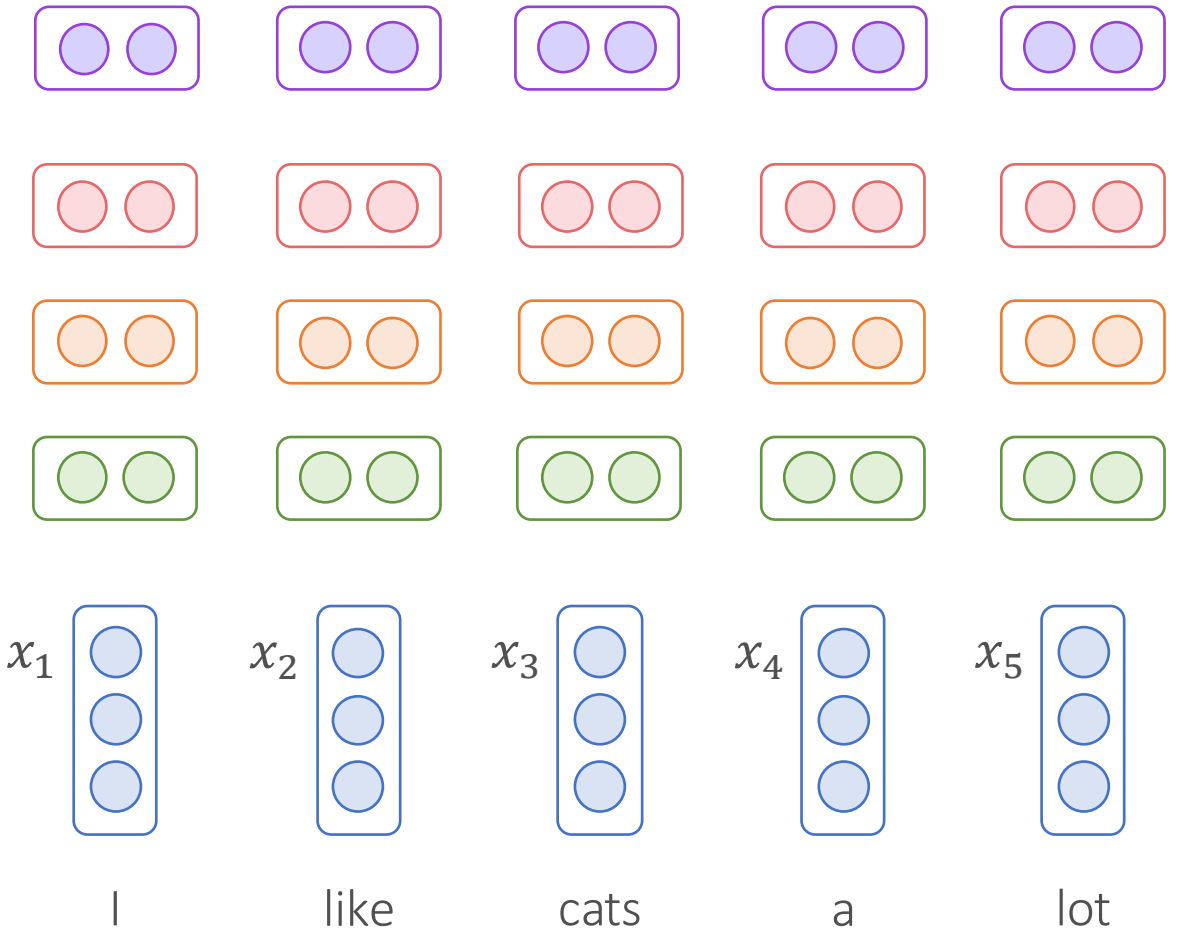
Single-Head Attention

Self-Attention Output

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Multi-Head Attention

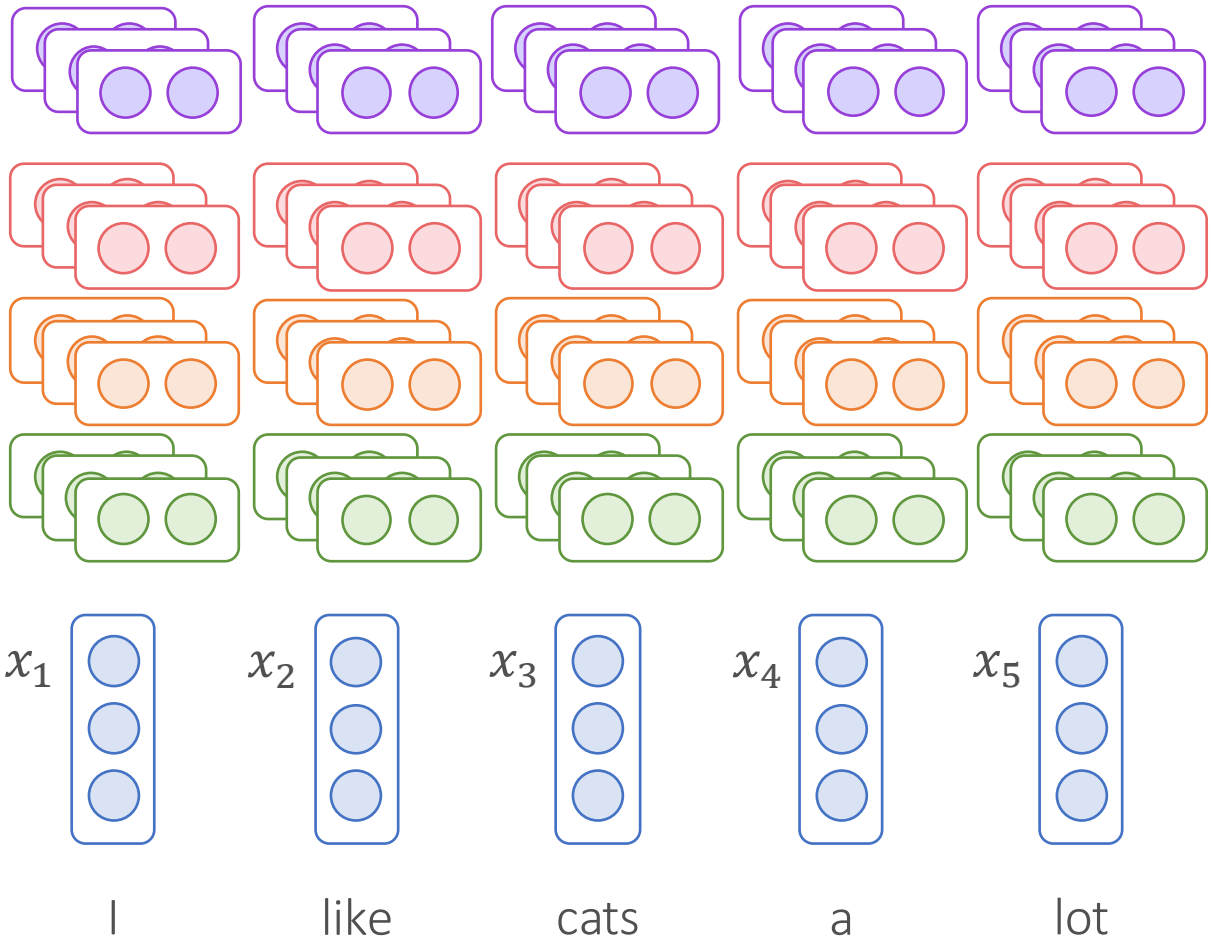
Each attention head focuses on different parts of understanding!

Multi-Attention Output

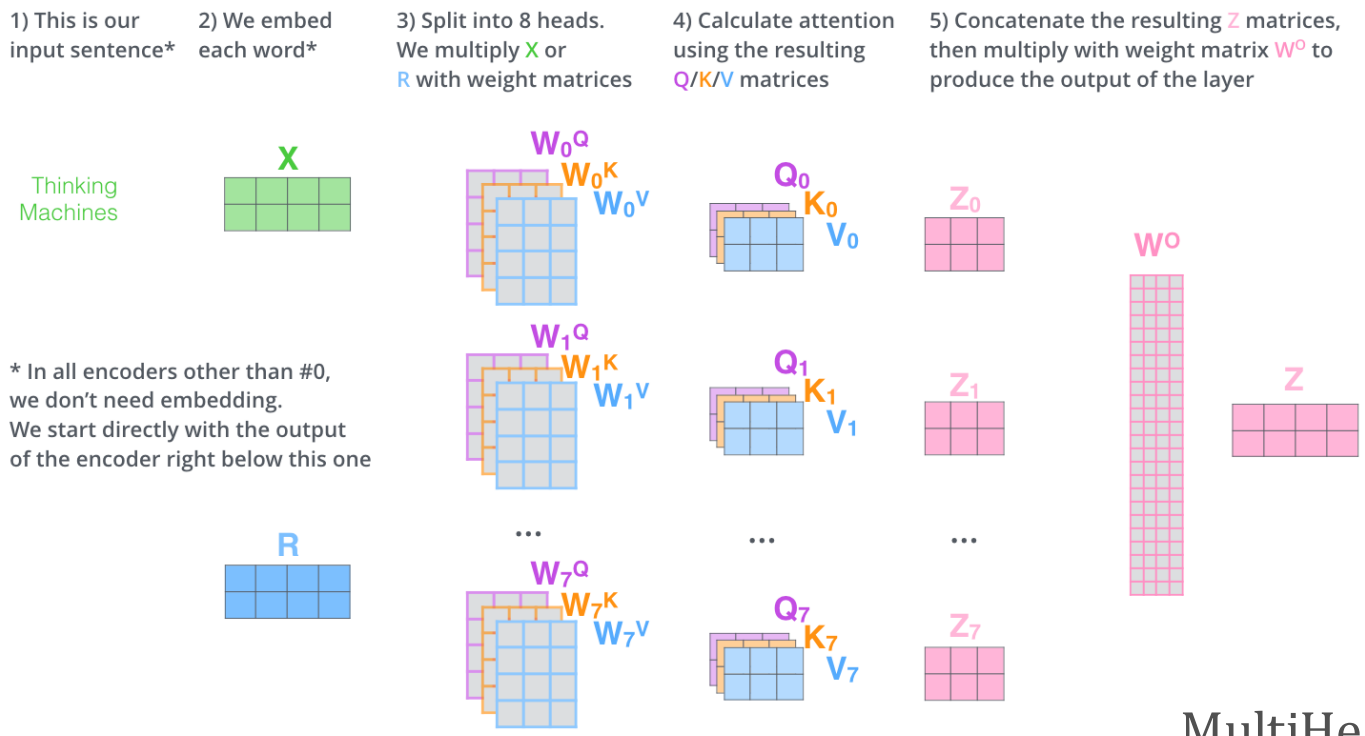
Query $q_i = W_j^Q x_i$

Key $k_i = W_j^K x_i$

Value $v_i = W_j^V x_i$



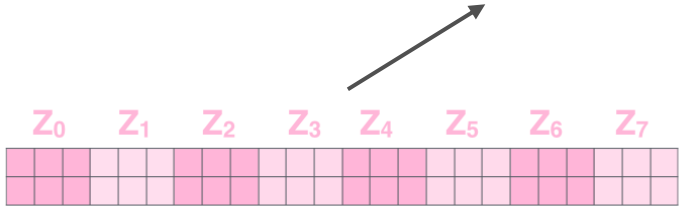
Multi-Head Attention – Matrix Form



$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

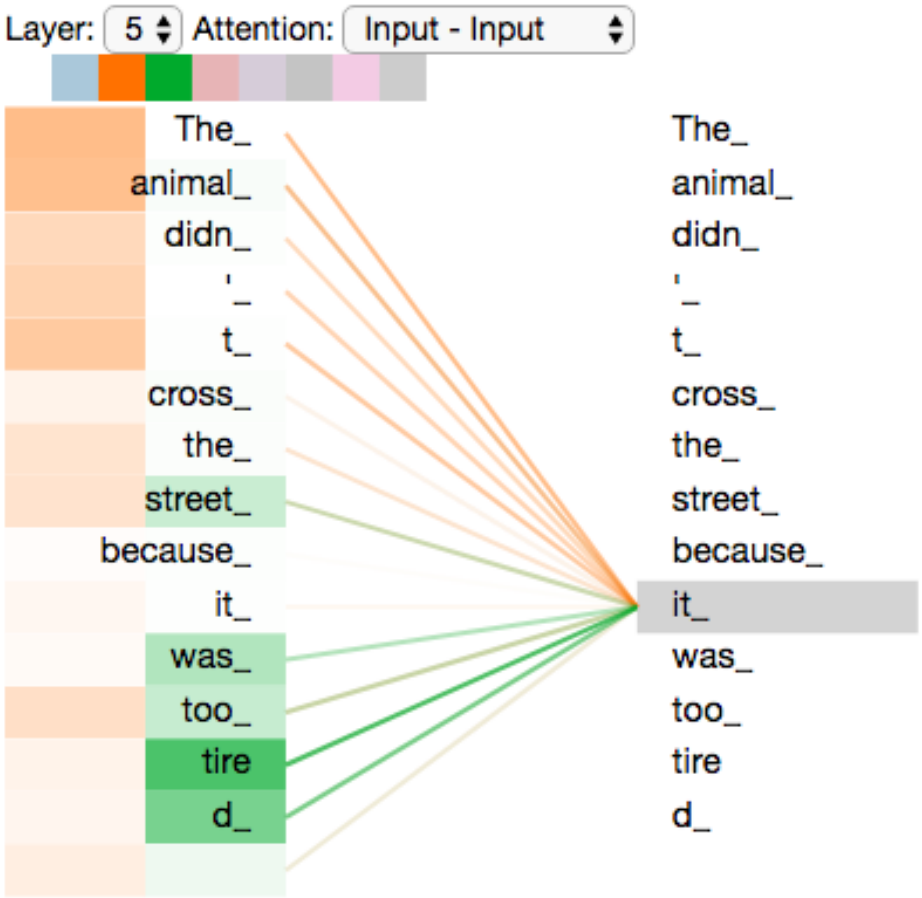
$$\text{head}_i = \text{Attention}(XW_i^Q, XW_i^K, XW_i^V)$$

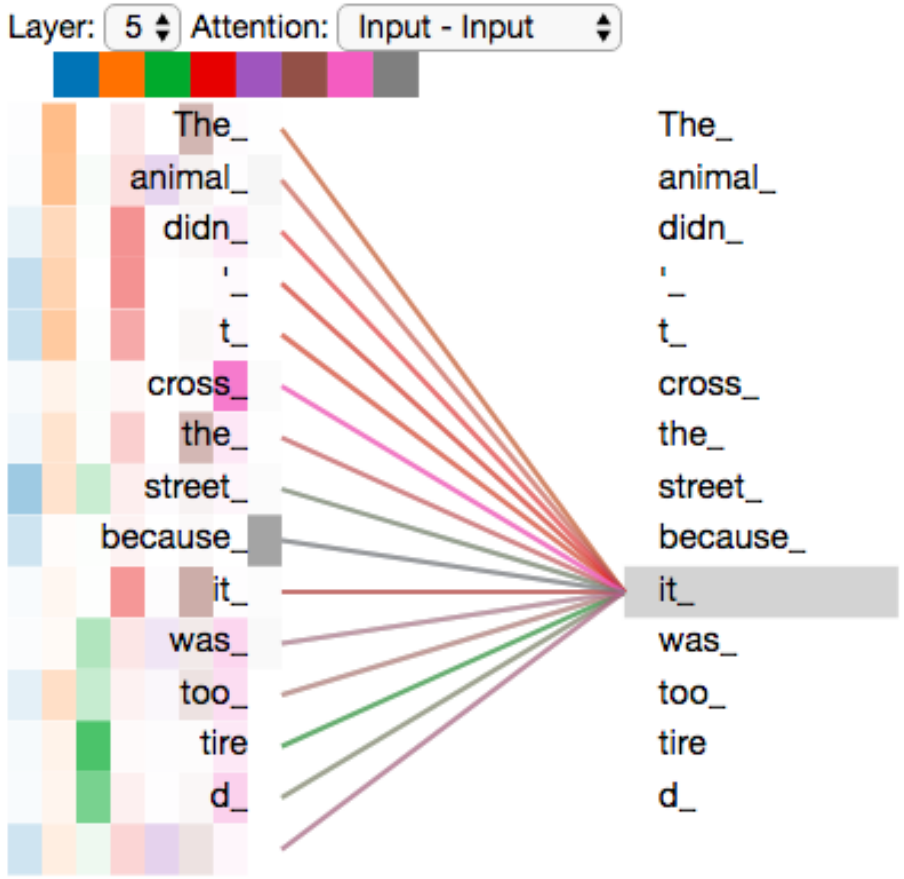
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$



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

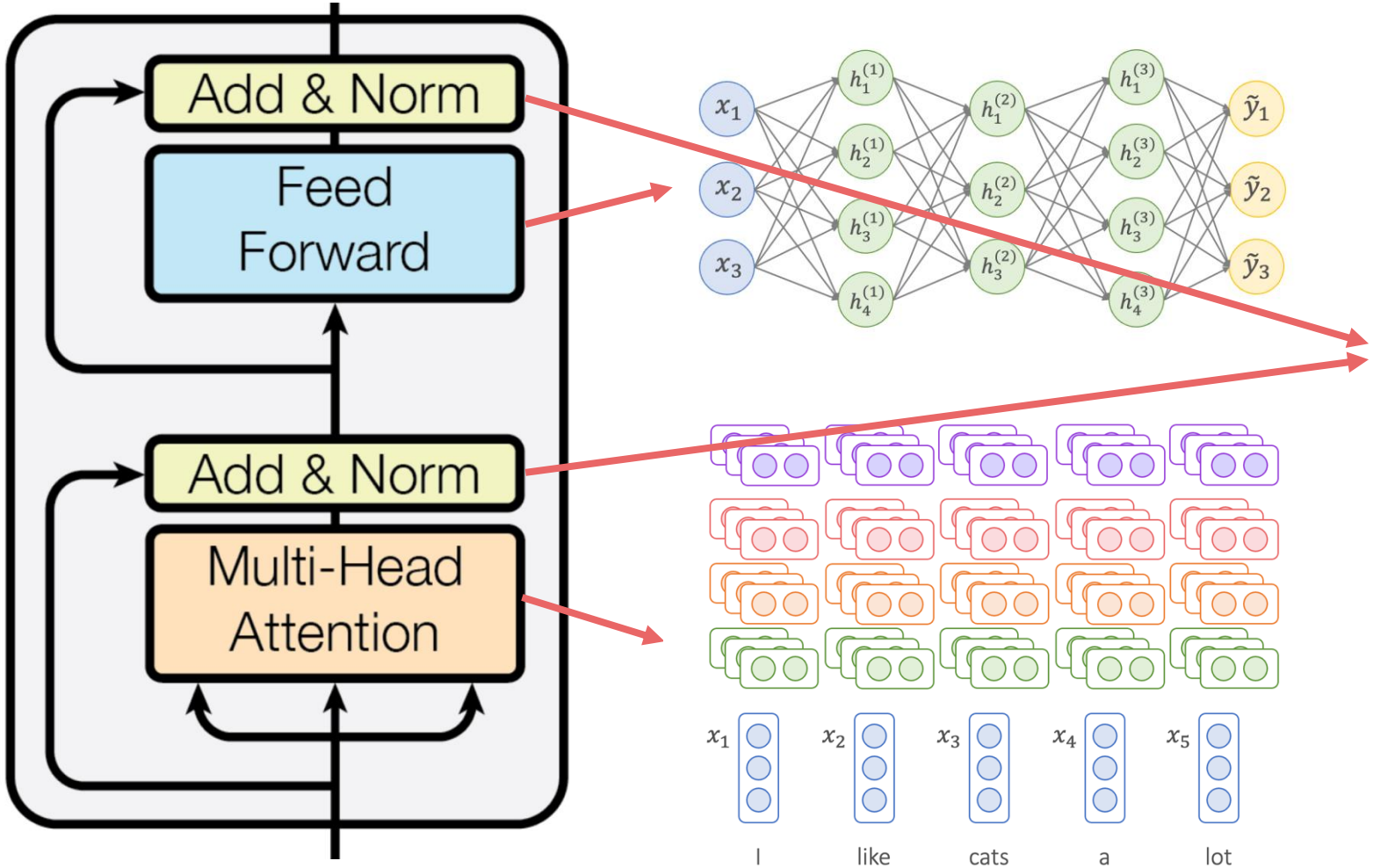
What Does Multi-Head Attention Learn?





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

Transformer Layer

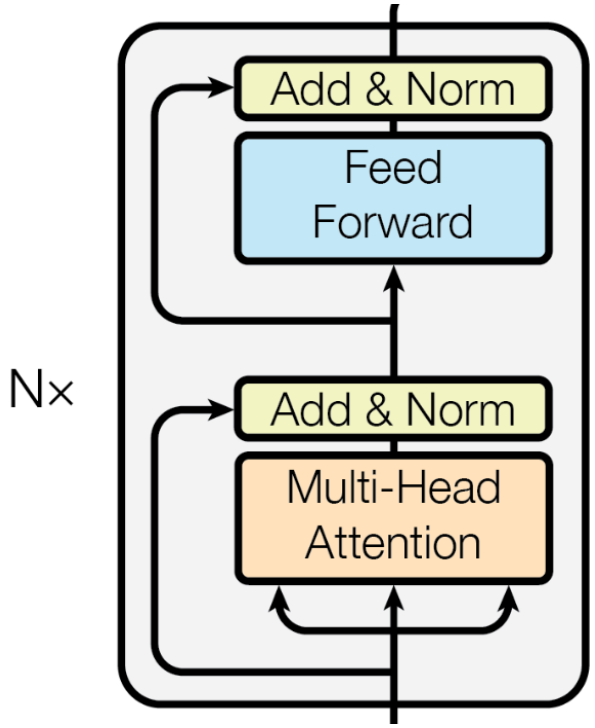


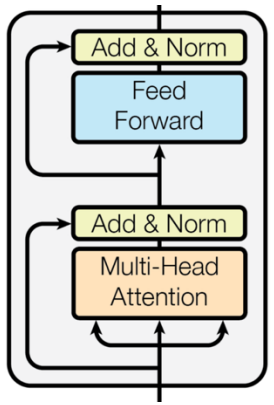
$$\text{LayerNorm}(x + \text{Sublayer}(x))$$

$$y = \frac{x - \mathbf{E}[x]}{\sqrt{\mathbf{Var}[x] + \epsilon}} * \gamma + \beta$$

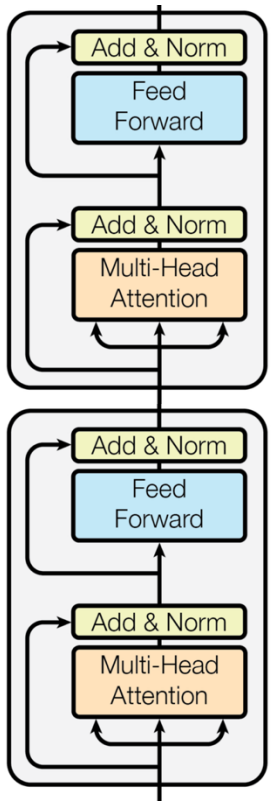
Residual connection (He et al., 2016)
Layer normalization (Ba et al., 2016)

Transformer Encoder

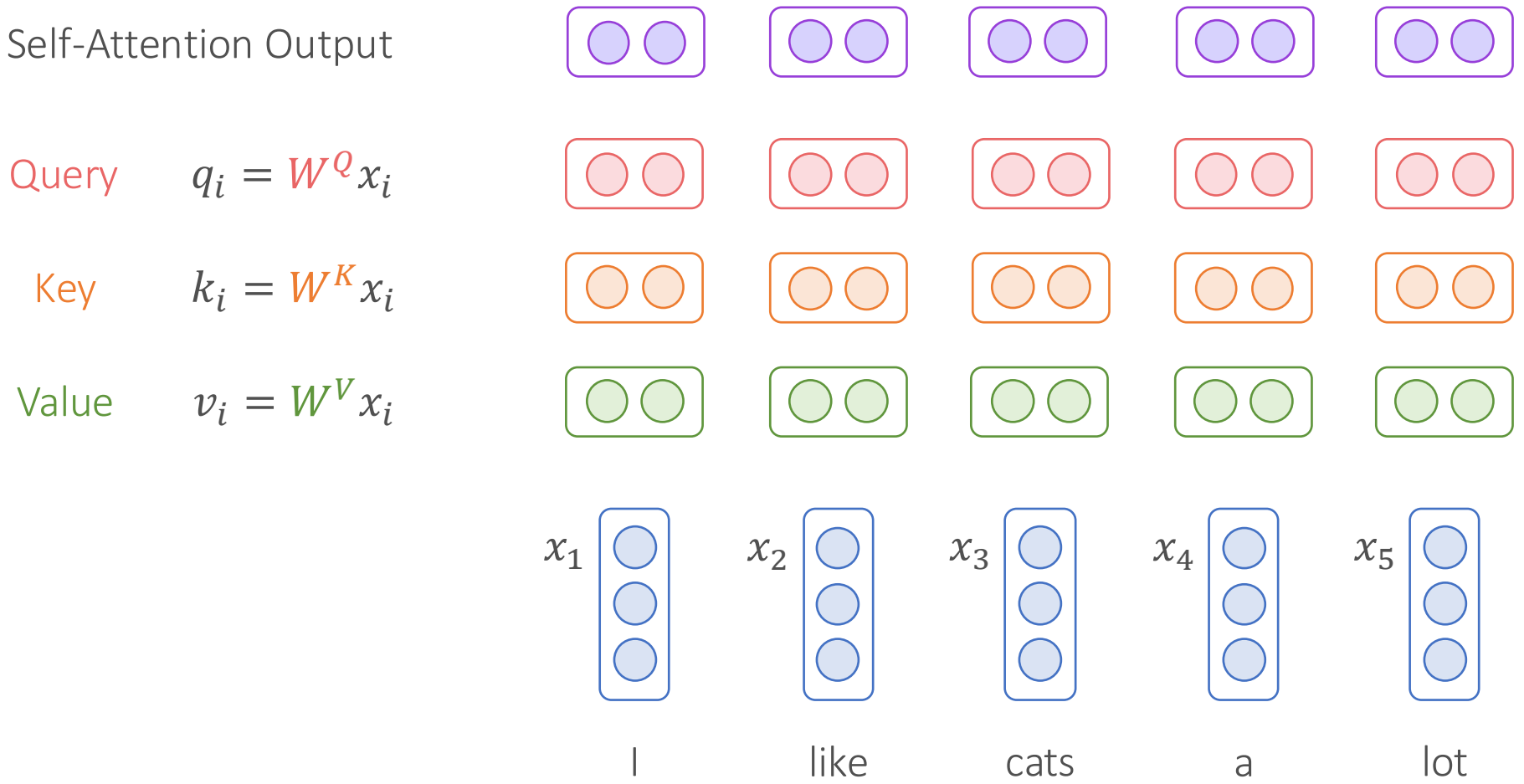




...



How About Word Order?



How About Word Order?

Weighted Sum

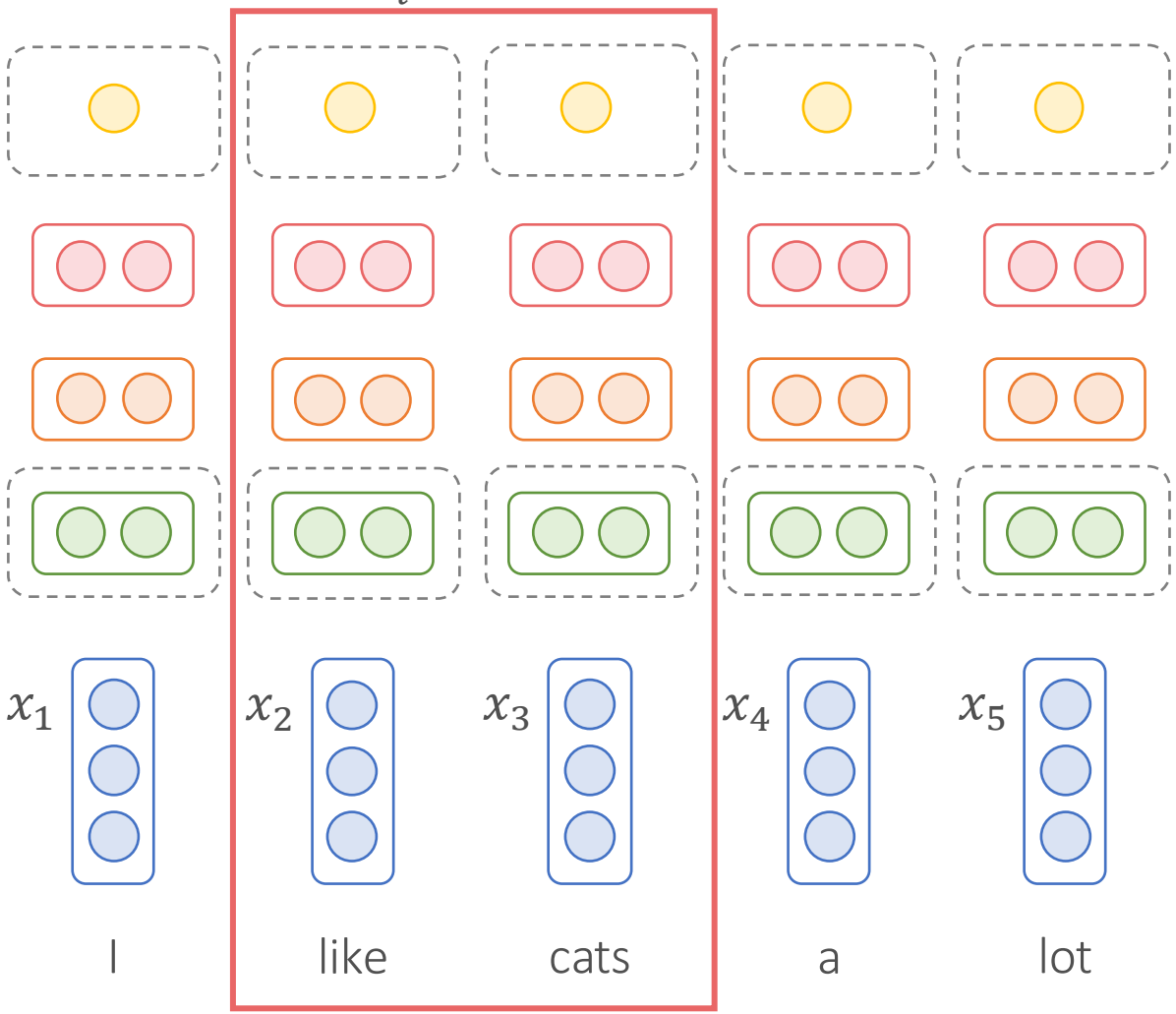
$z_1 = \sum_i \alpha_{1,i} v_i$

Normalized Attention Scores

Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



How About Word Order?

Weighted Sum

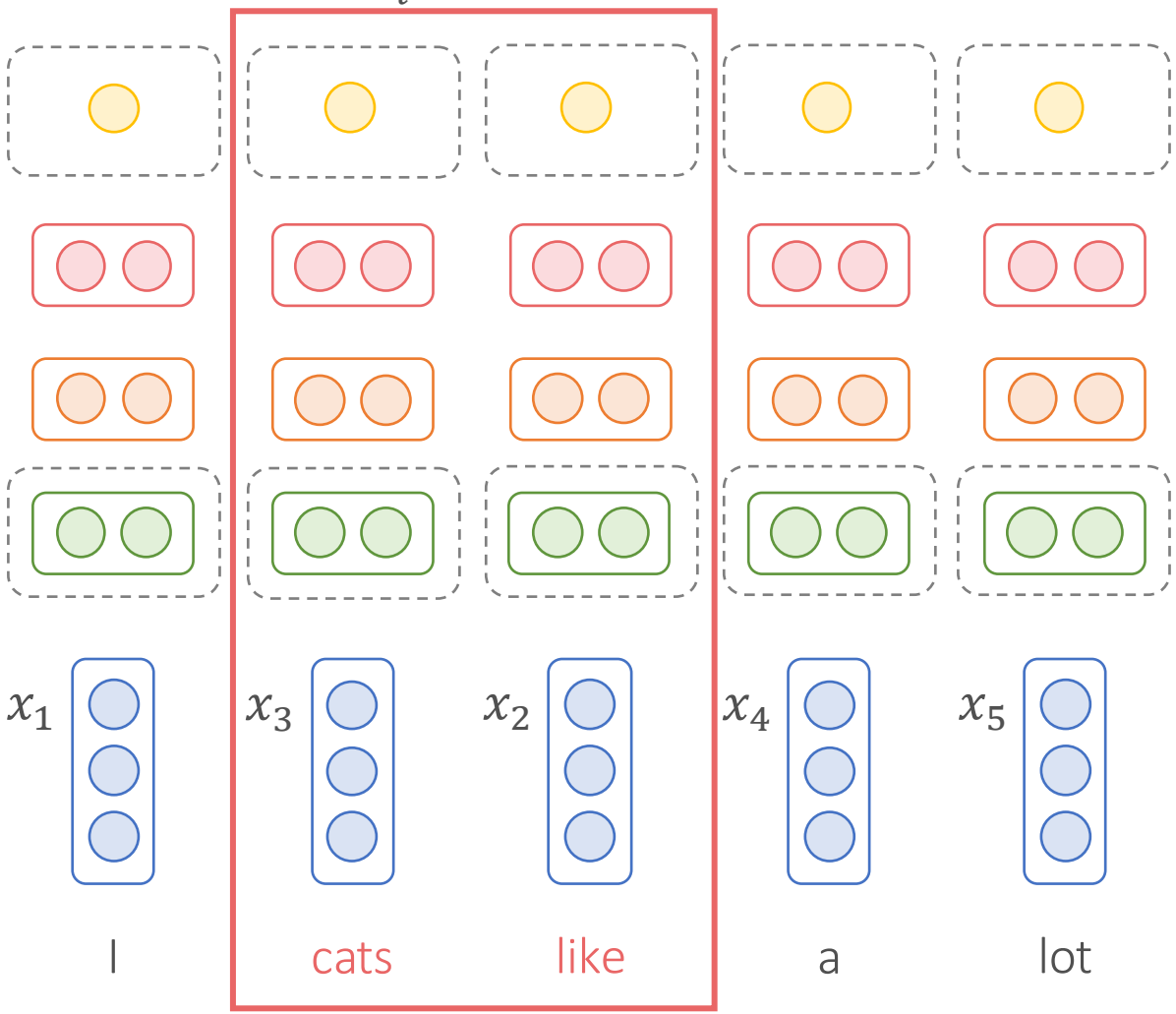
$z_1 = \sum_i \alpha_{1,i} v_i$

Normalized Attention Scores

Query $q_i = W^Q x_i$

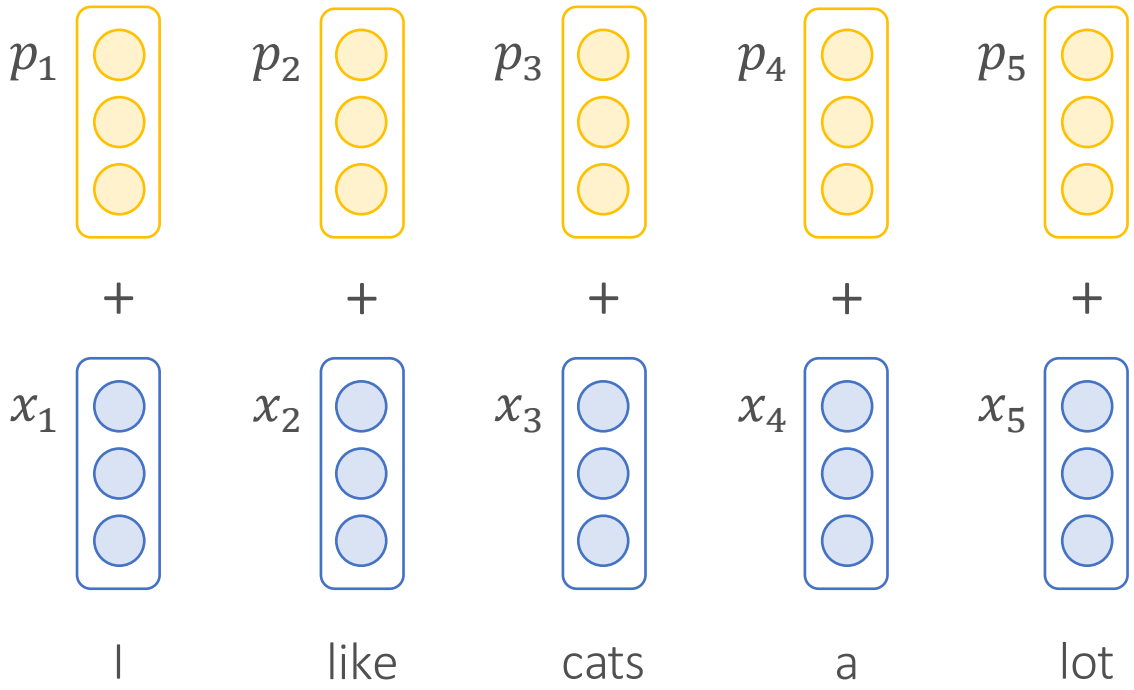
Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



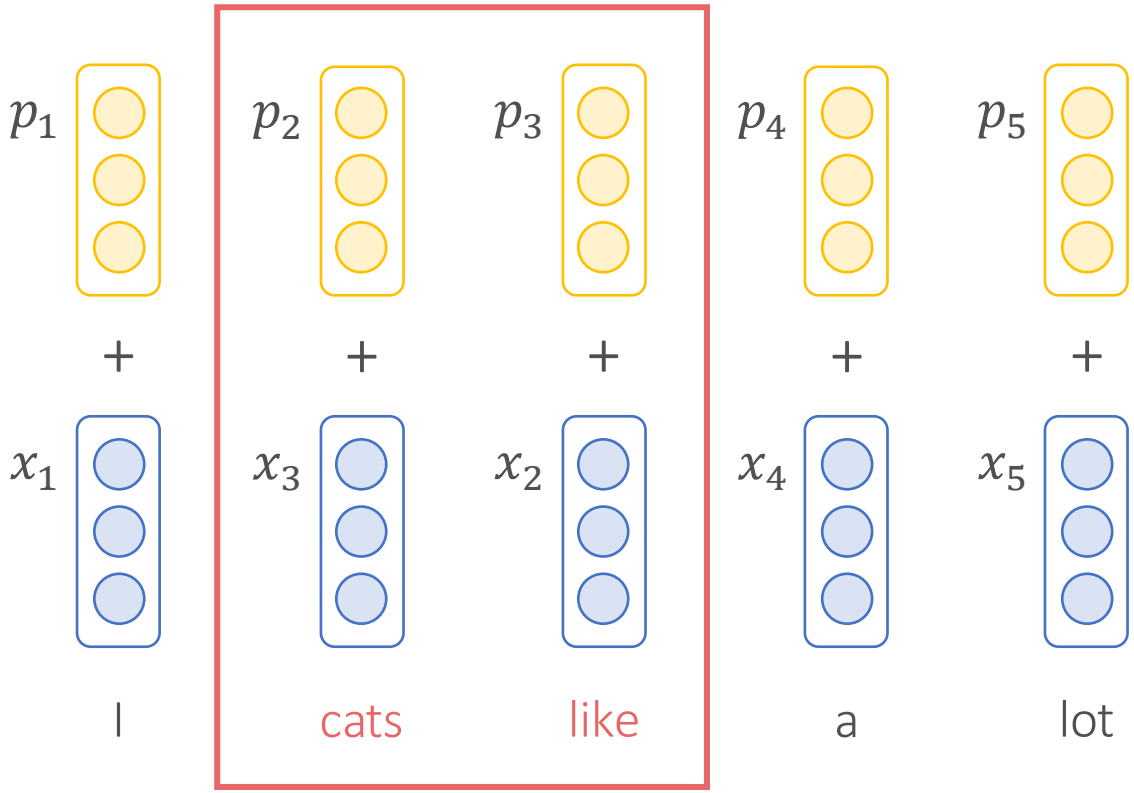
Solution: Positional Encoding

$$x_i \leftarrow x_i + PE_i$$



Solution: Positional Encoding

$$x_i \leftarrow x_i + PE_i$$



Solution: Positional Encoding

- Unique encoding for each position
- Closer positions should have more similar encodings
- Distance between neighboring positions should be the same

Sinusoidal Positional Encoding

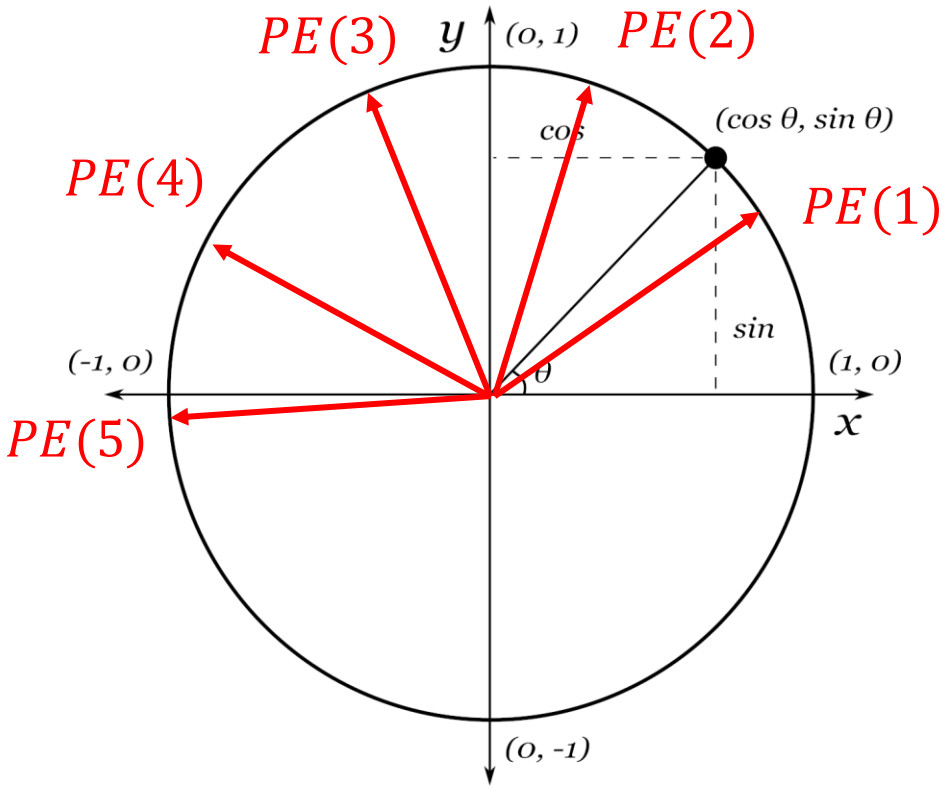
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Why this?

Sinusoidal Positional Encoding: Intuition

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

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

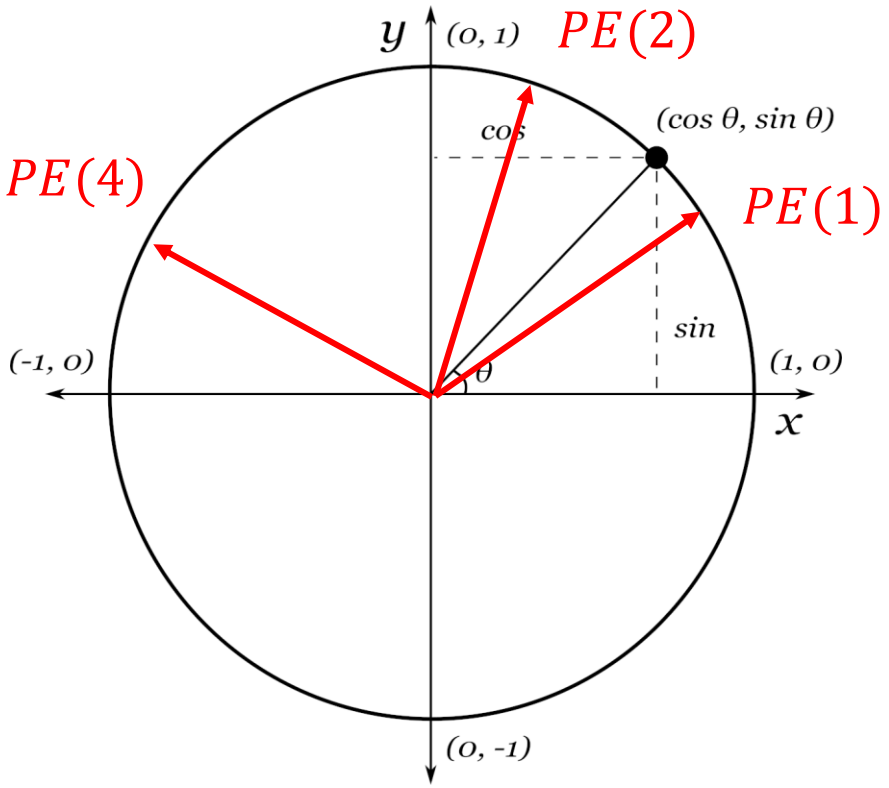


https://www.inchcalculator.com/unit-circle-calculator/

Sinusoidal Positional Encoding: Intuition

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

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



$$Cosine\big(PE(1), PE(2)\big) > Cosine(PE(1), PE(4))$$

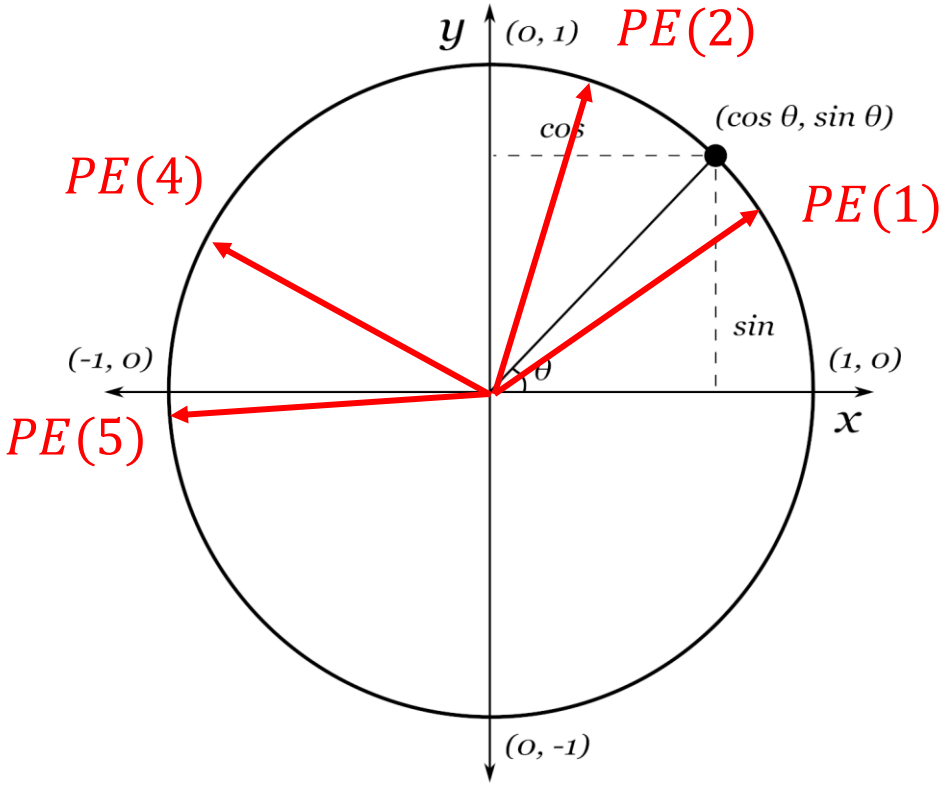
Closer positions should have more similar encodings

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

Sinusoidal Positional Encoding: Intuition

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

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



$$Cosine\big(PE(1), PE(2)\big) = Cosine(PE(4), PE(5))$$

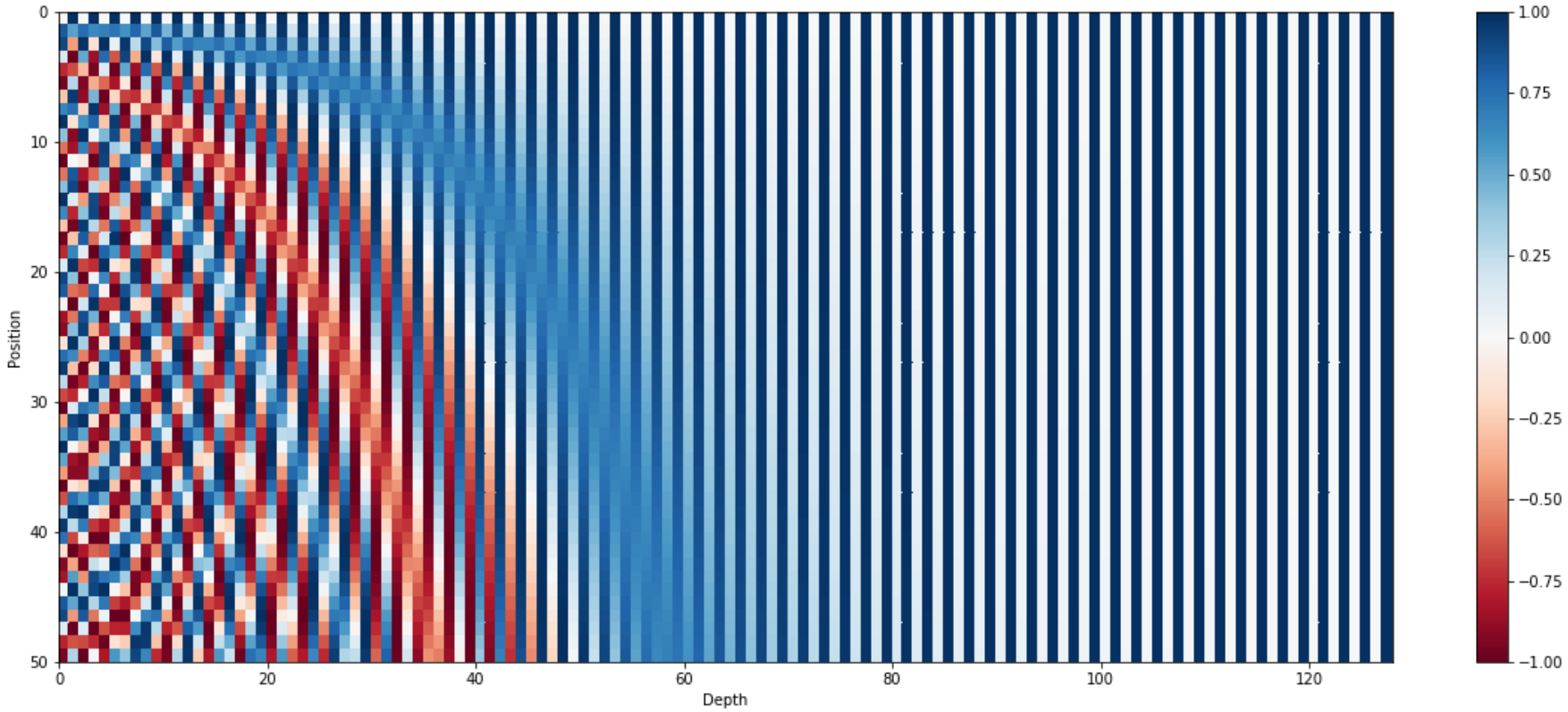
Distance between neighboring positions should be the same

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

Sinusoidal Positional Encoding

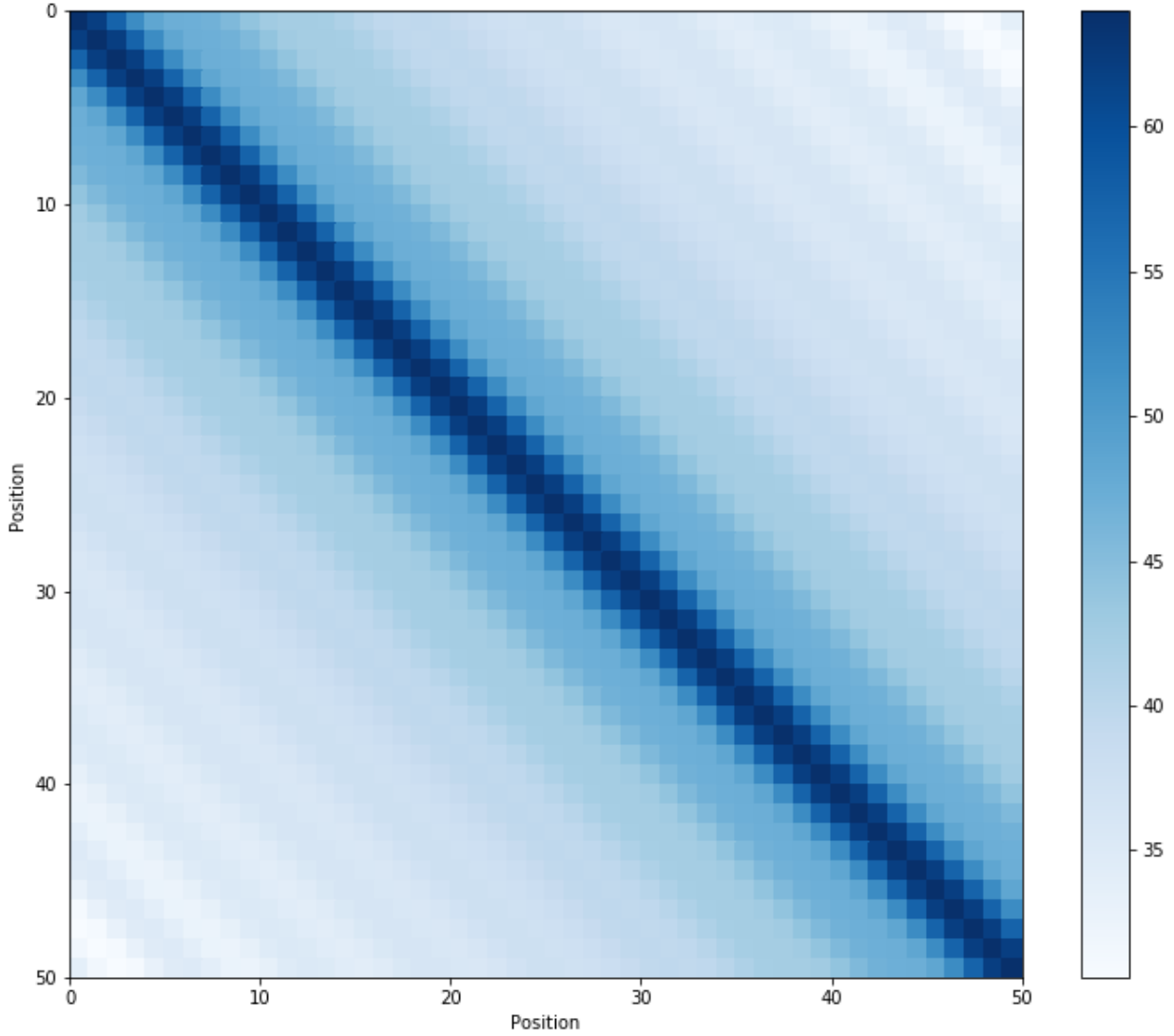
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

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



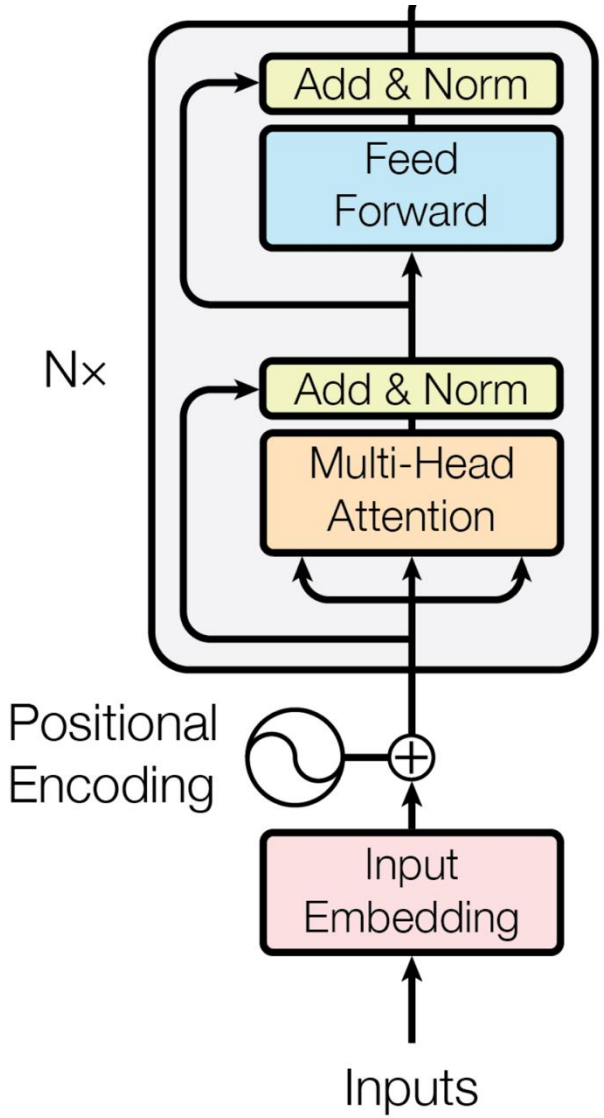
https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

Sinusoidal Positional Encoding



https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

Transformer with Positional Encoding

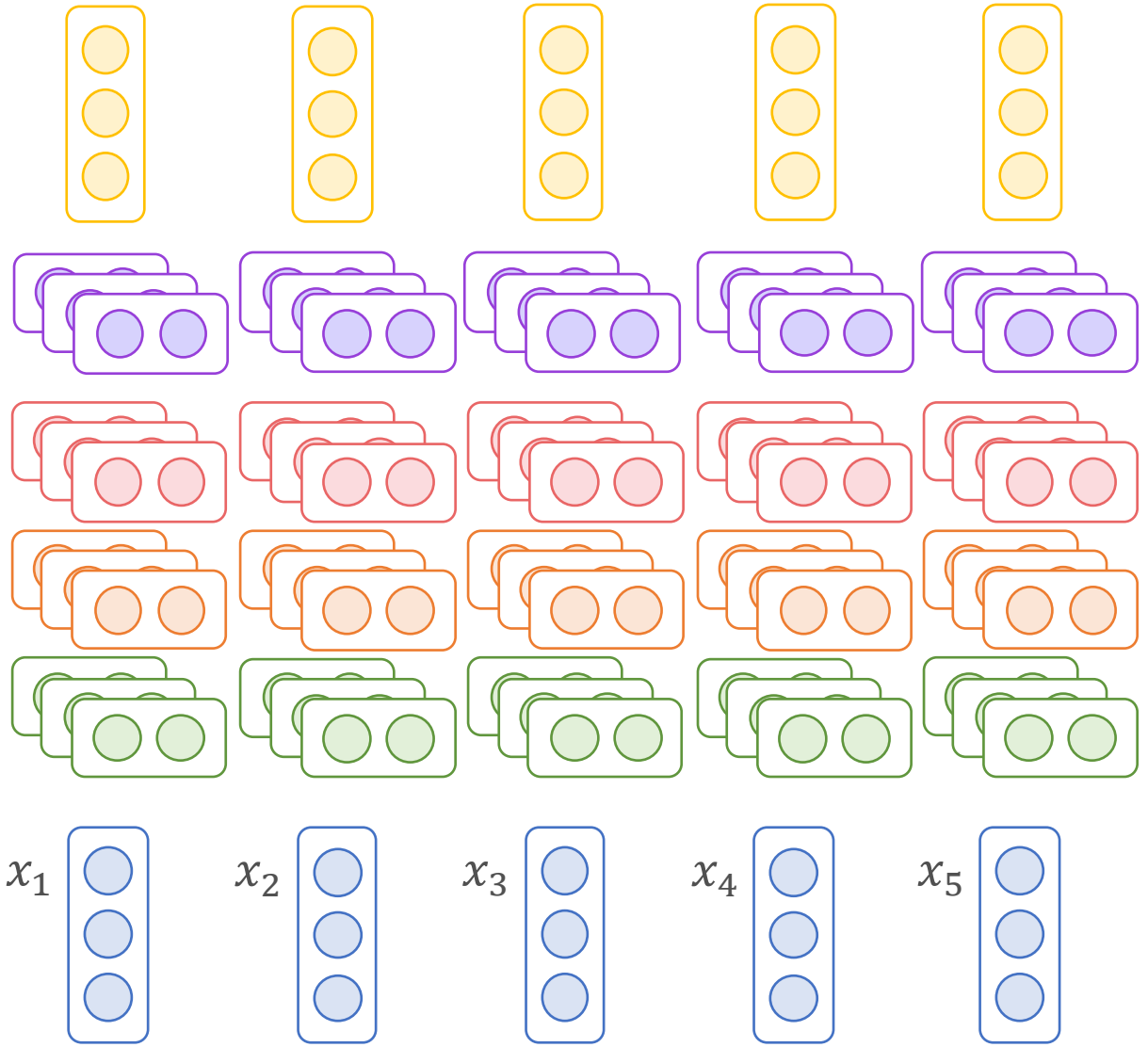


Transformer as Token-Level Encoder

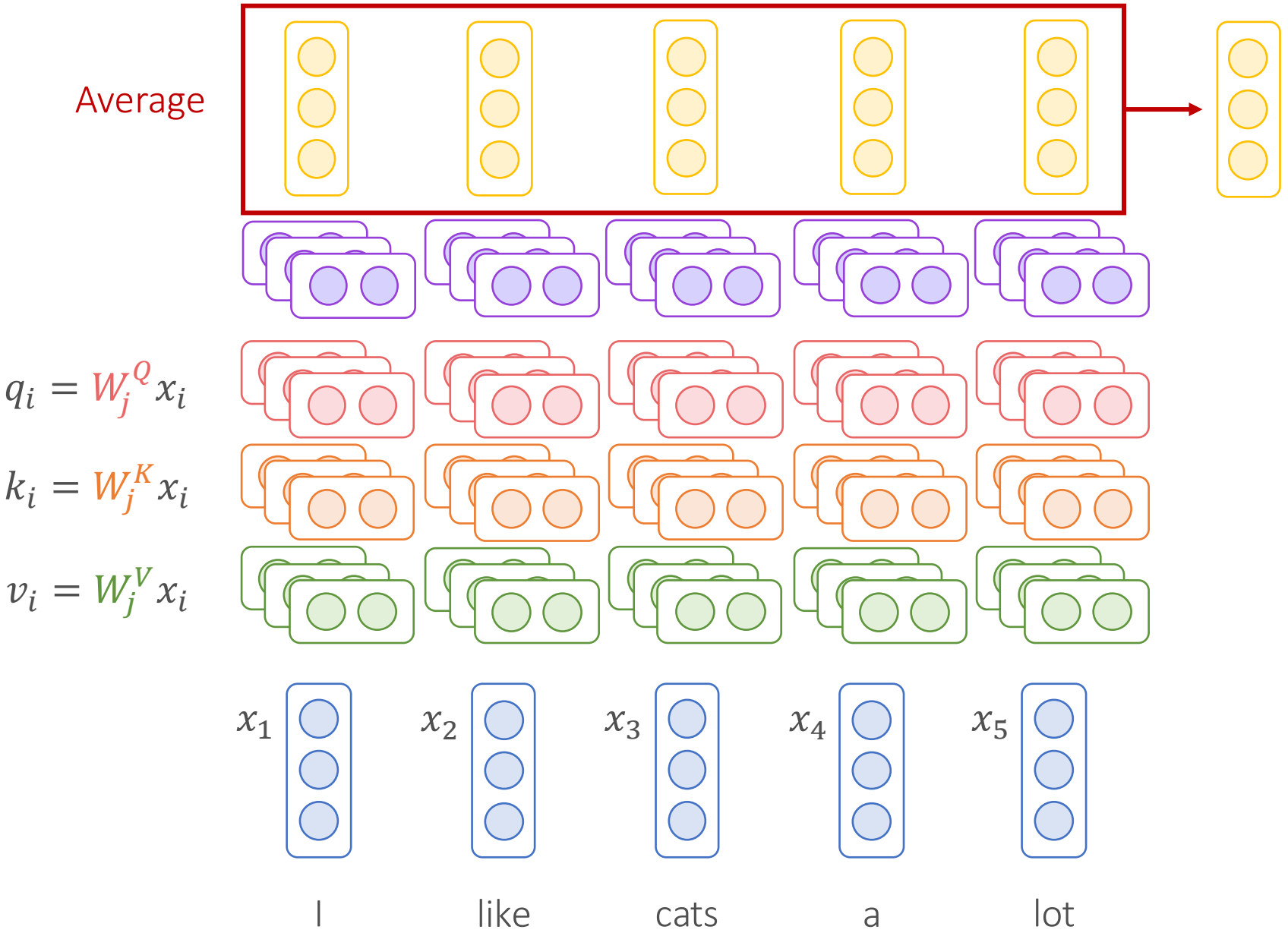
$q_i = W_j^Q x_i$

$k_i = W_j^K x_i$

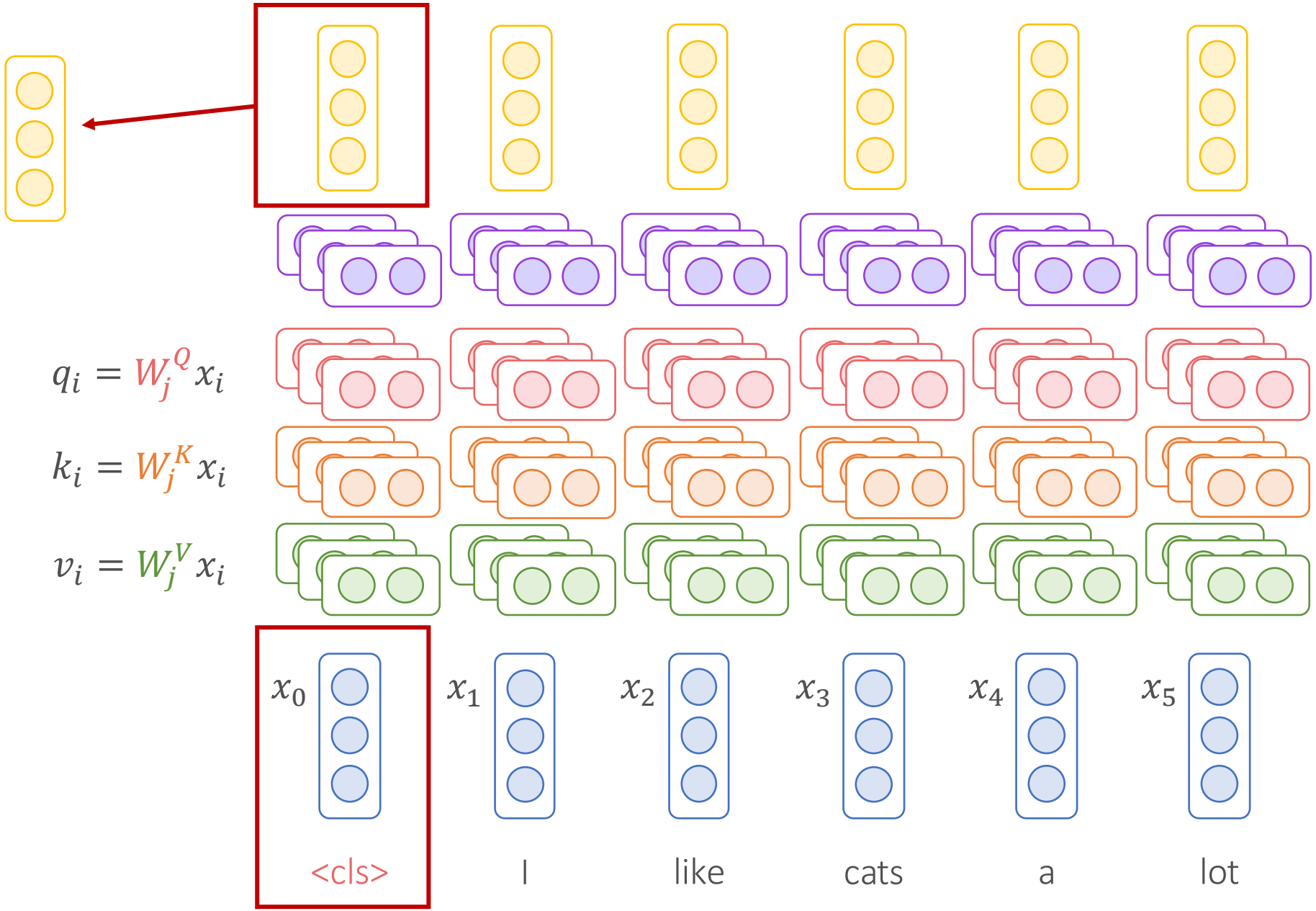
$v_i = W_j^V x_i$



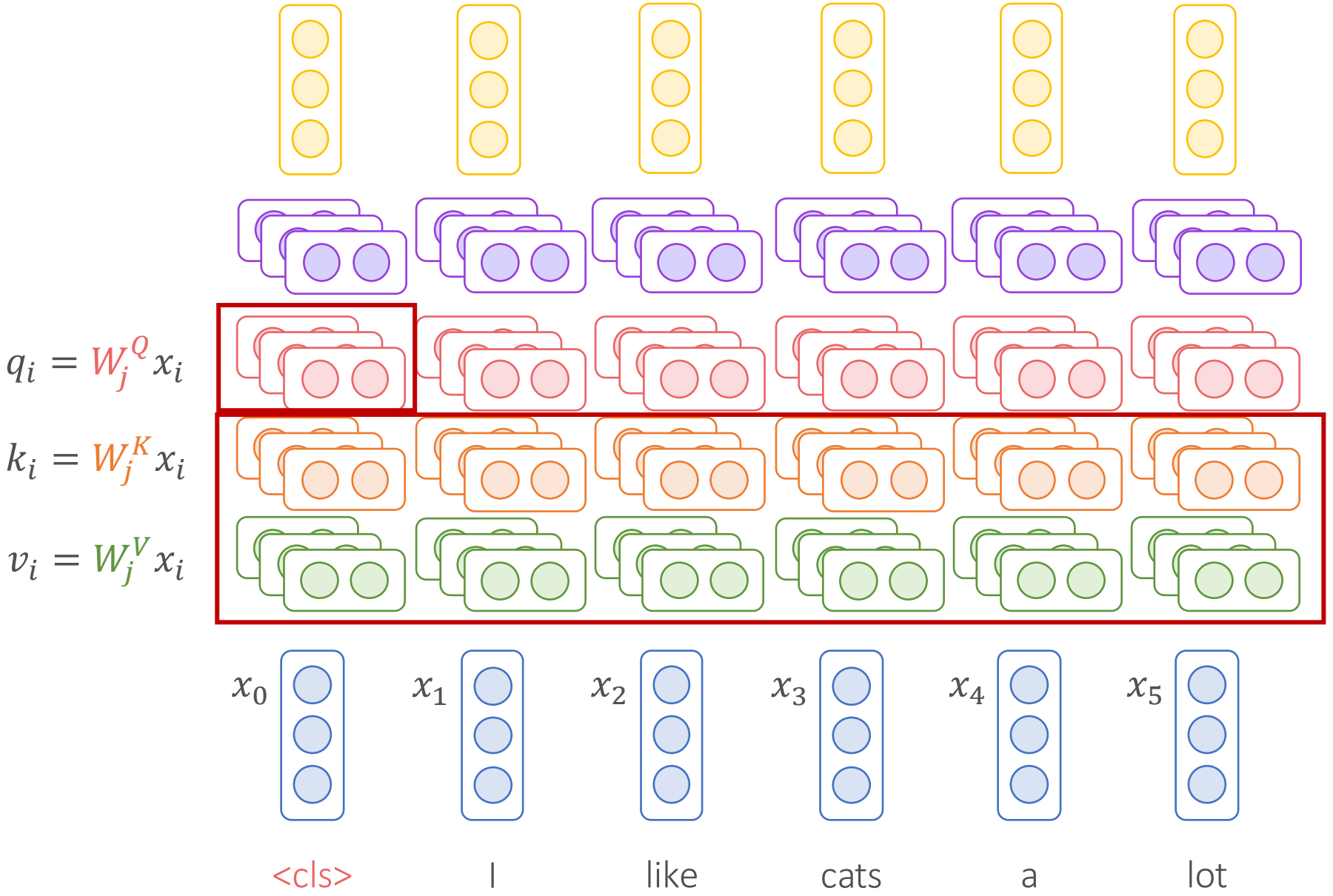
Transformer as Sentence-Level Encoder



Transformer as Sentence-Level Encoder



Transformer as Sentence-Level Encoder

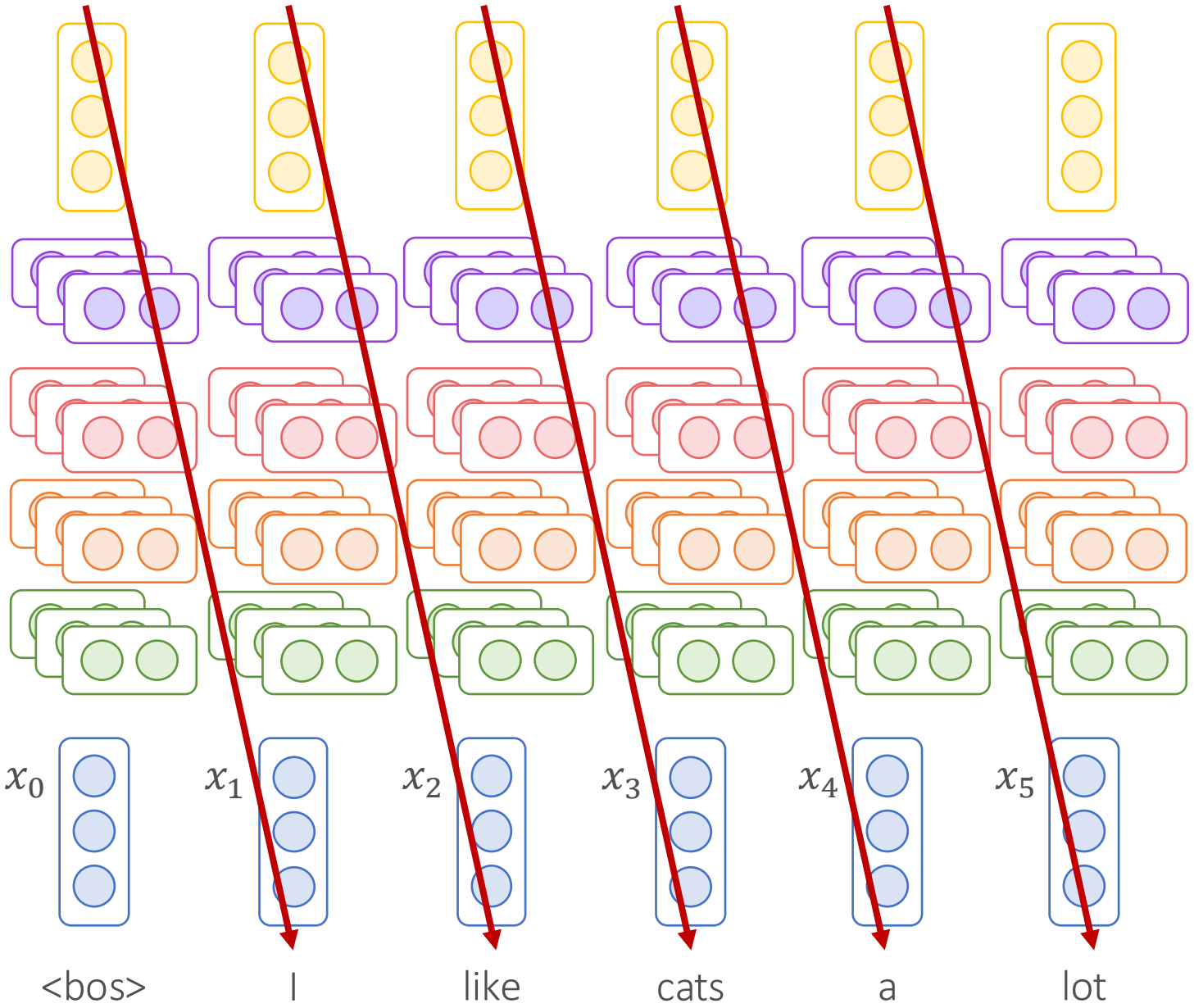


Transformer as Decoder?

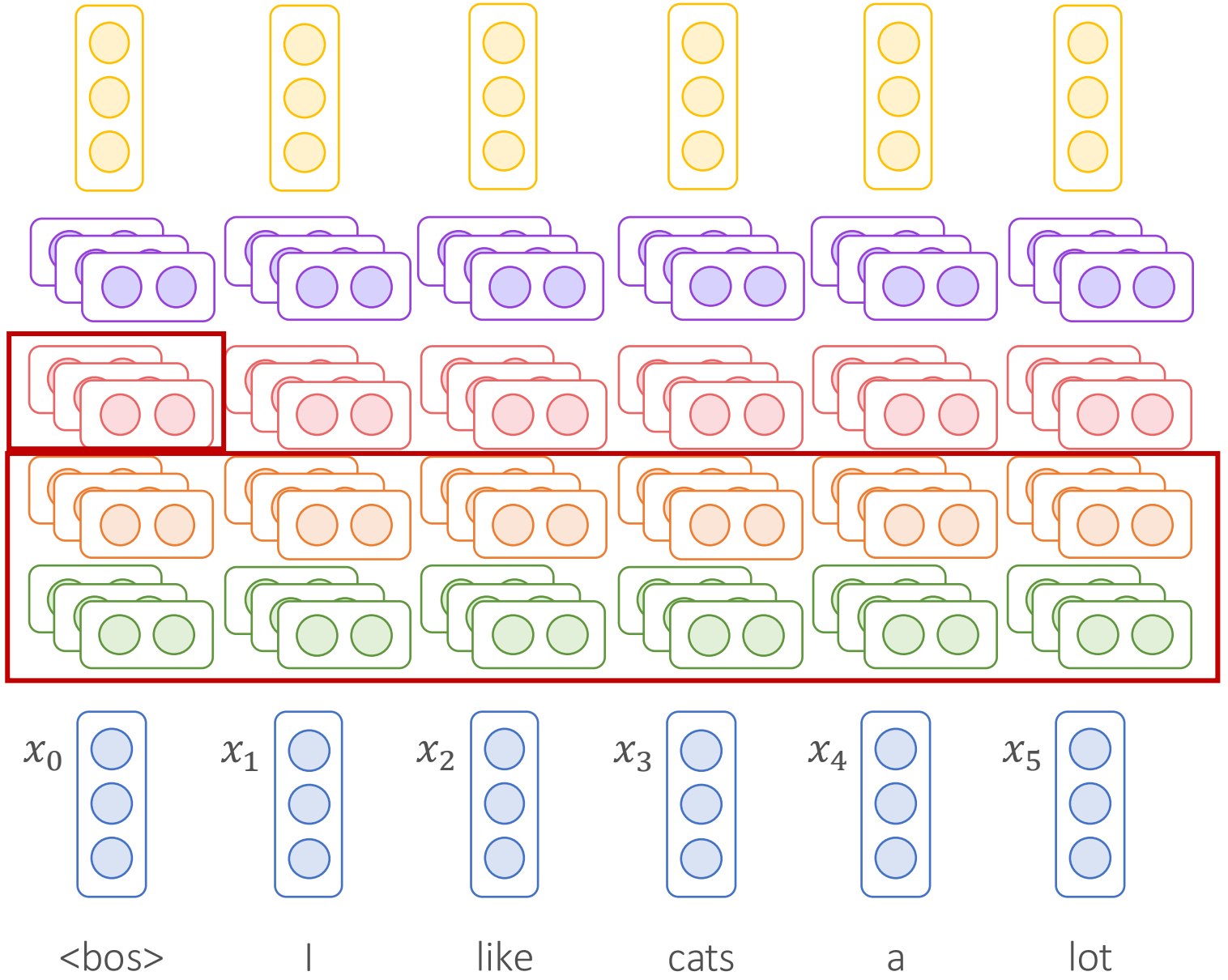
$q_i = W_j^Q x_i$

$k_i = W_j^K x_i$

$v_i = W_j^V x_i$

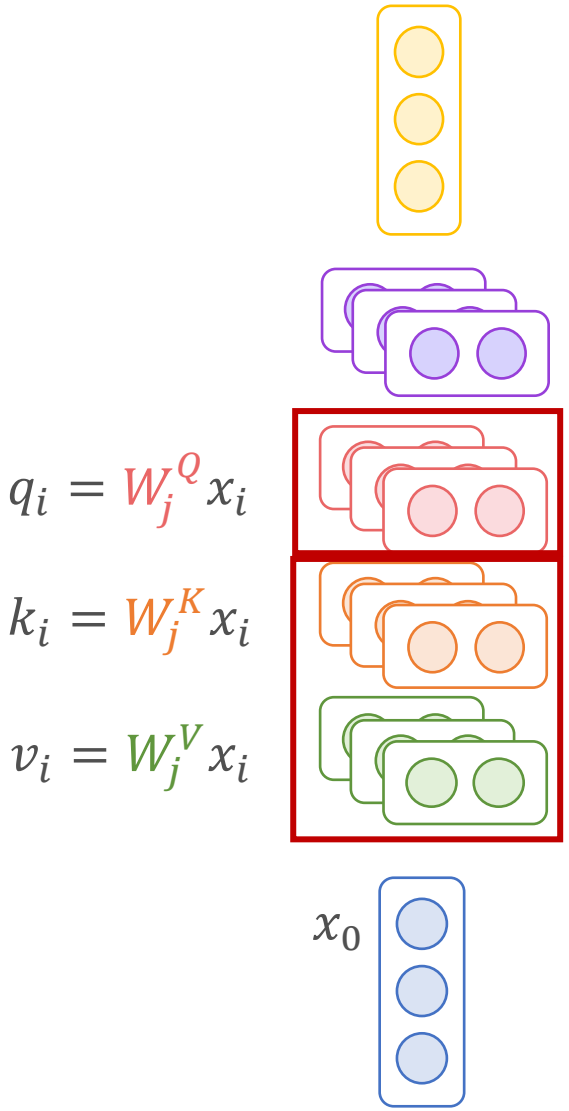


Transformer as Decoder?

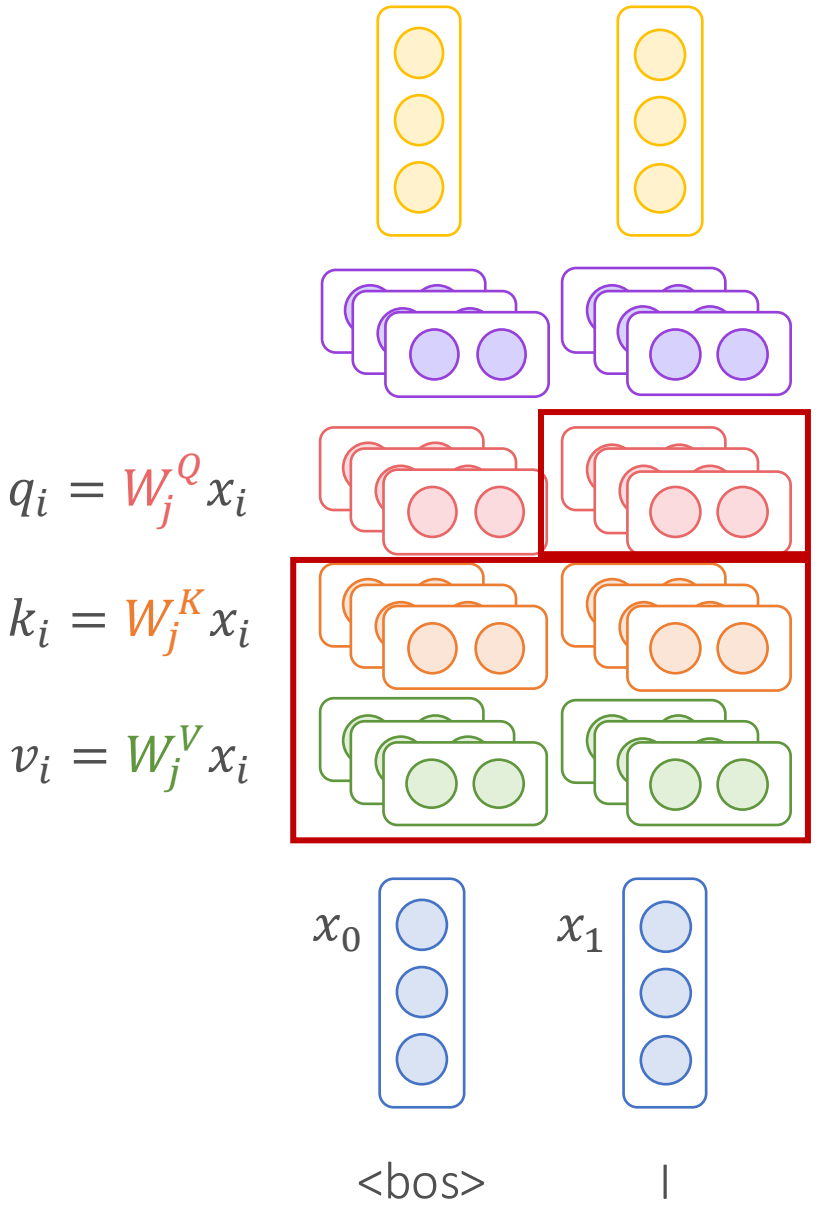


How to compute attention from the token we are going to generate?

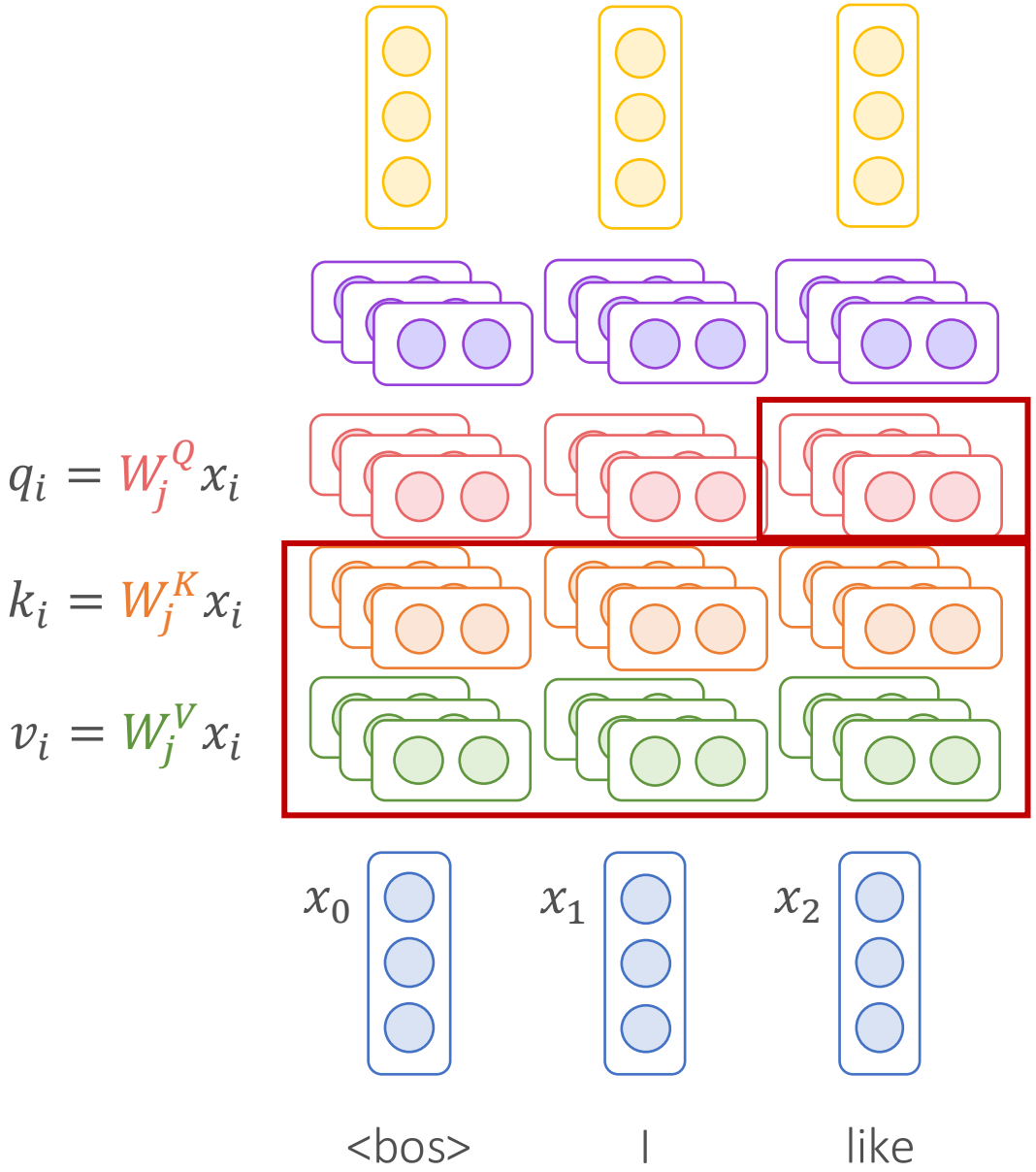
Transformer Decoder



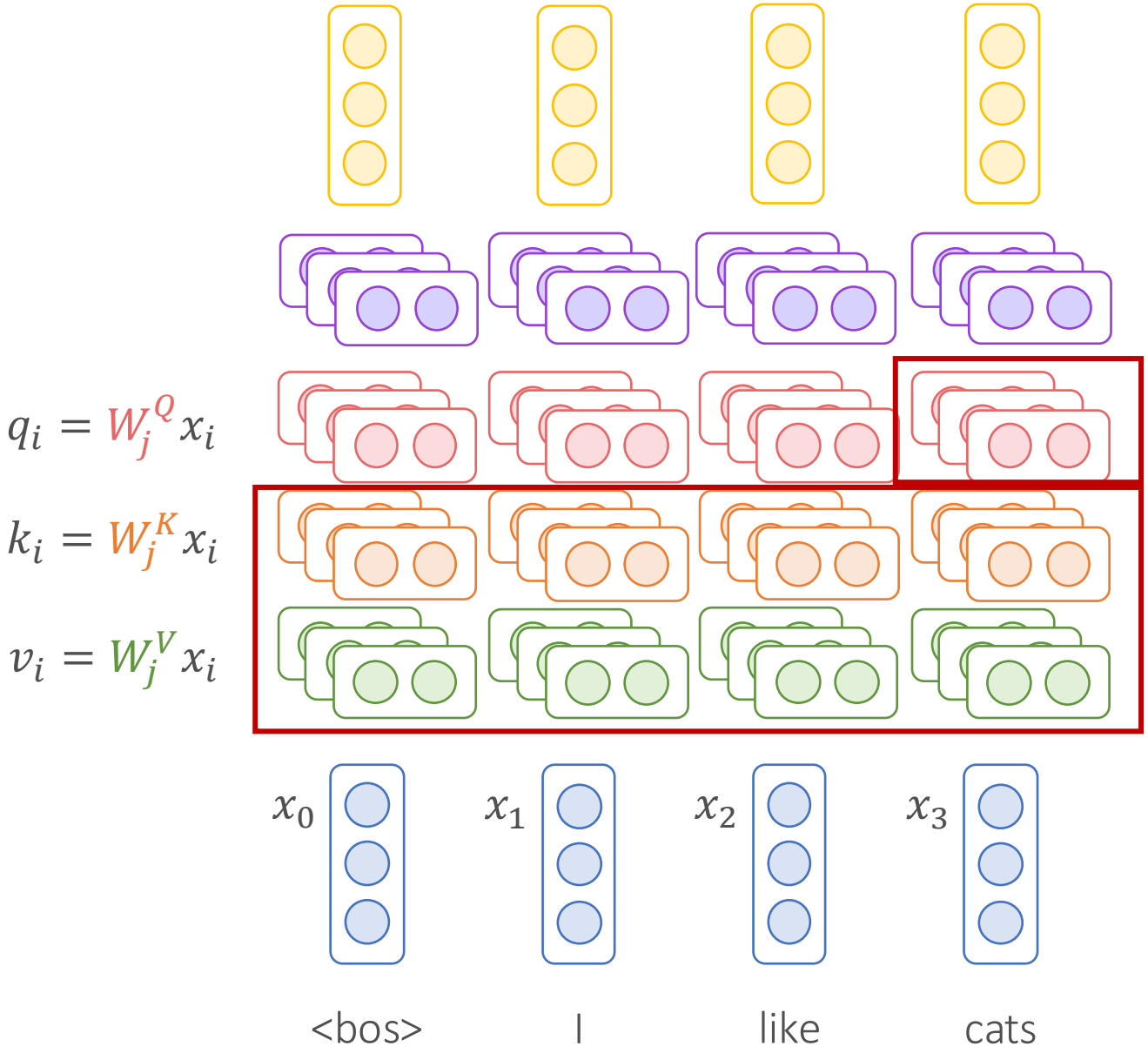
Transformer Decoder



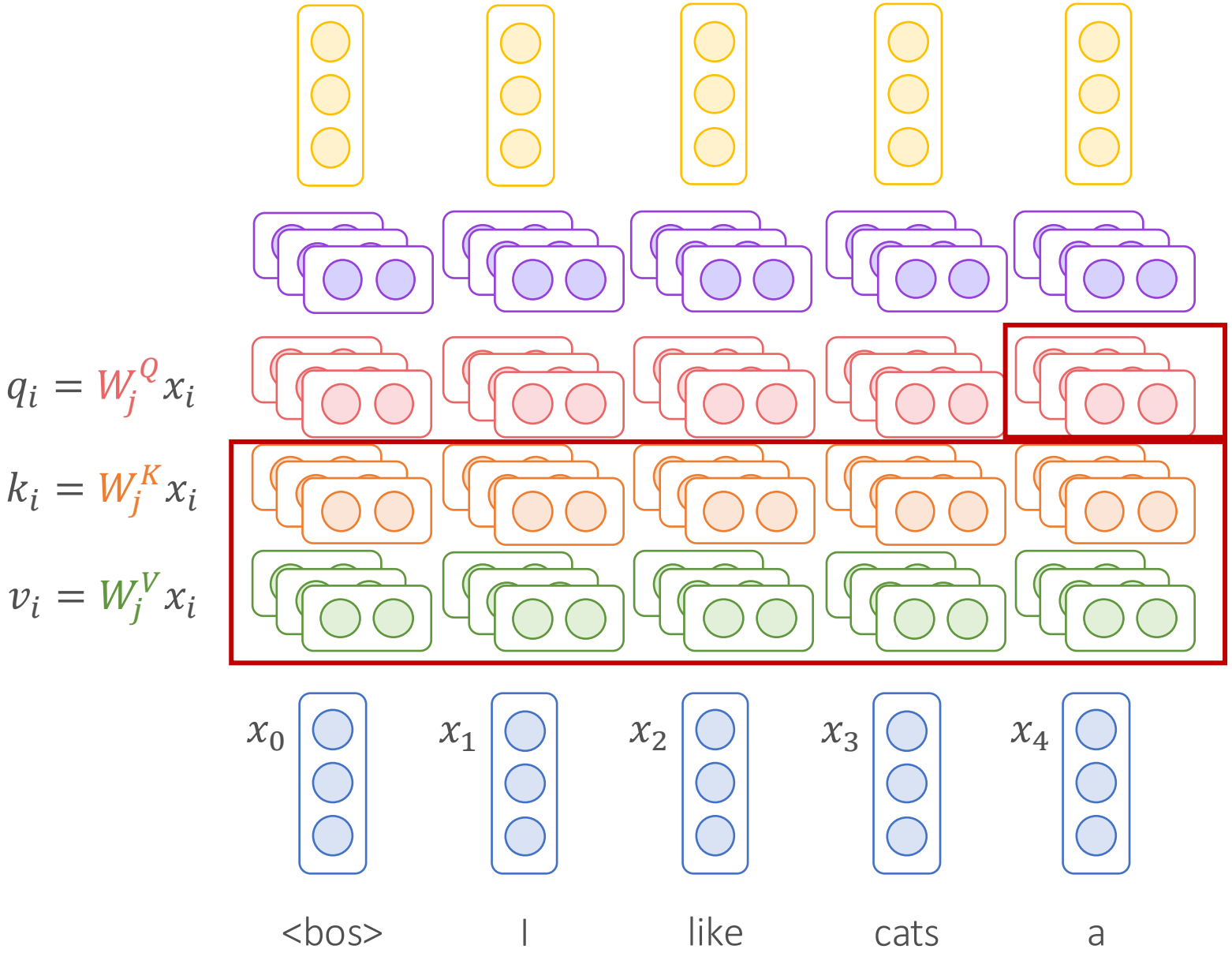
Transformer Decoder



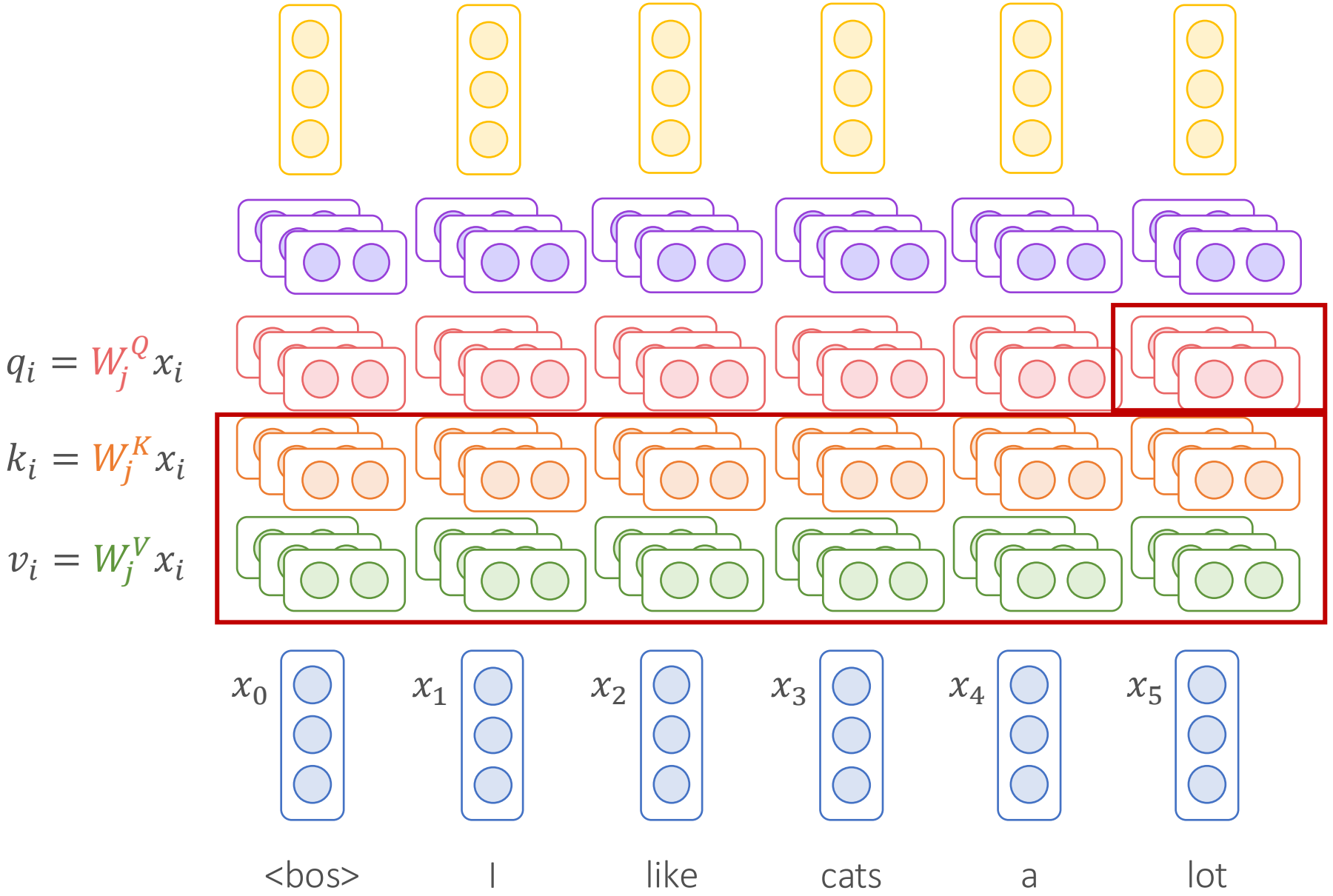
Transformer Decoder



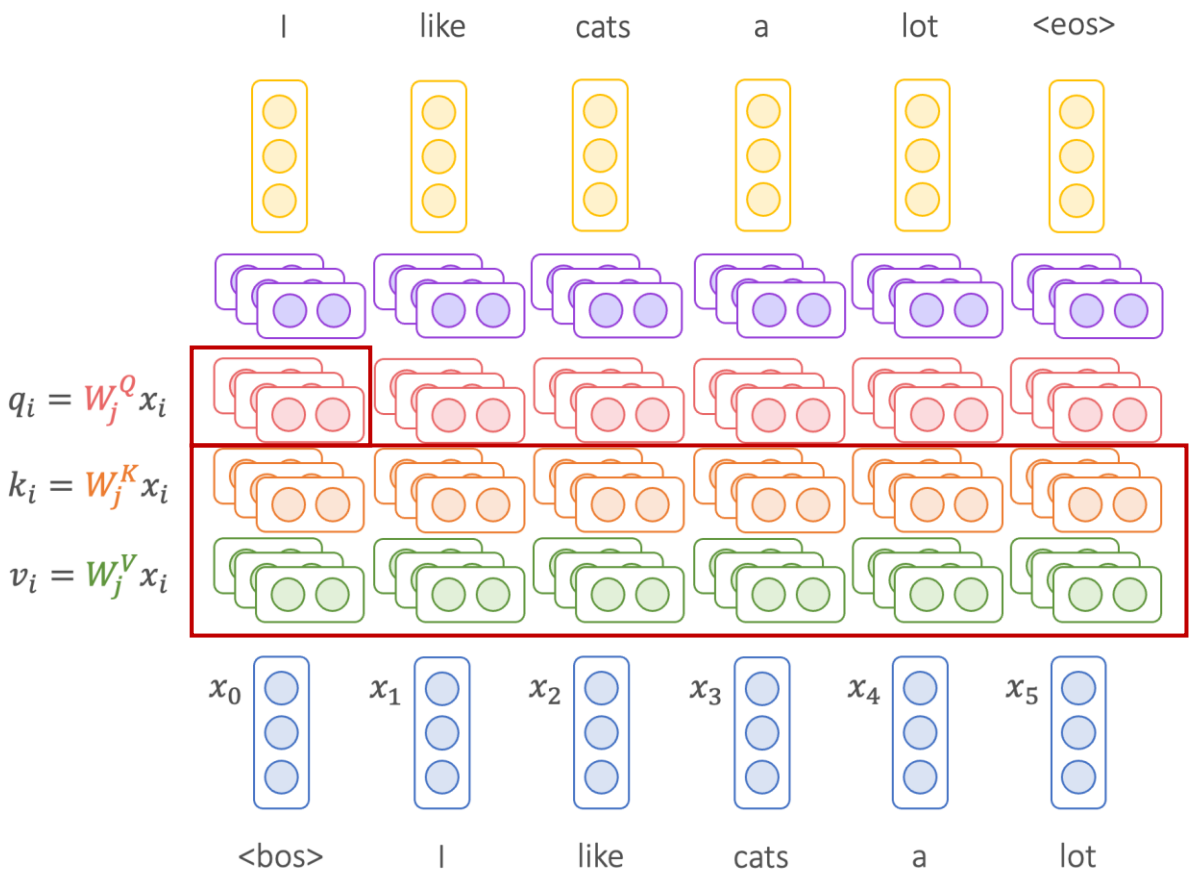
Transformer Decoder



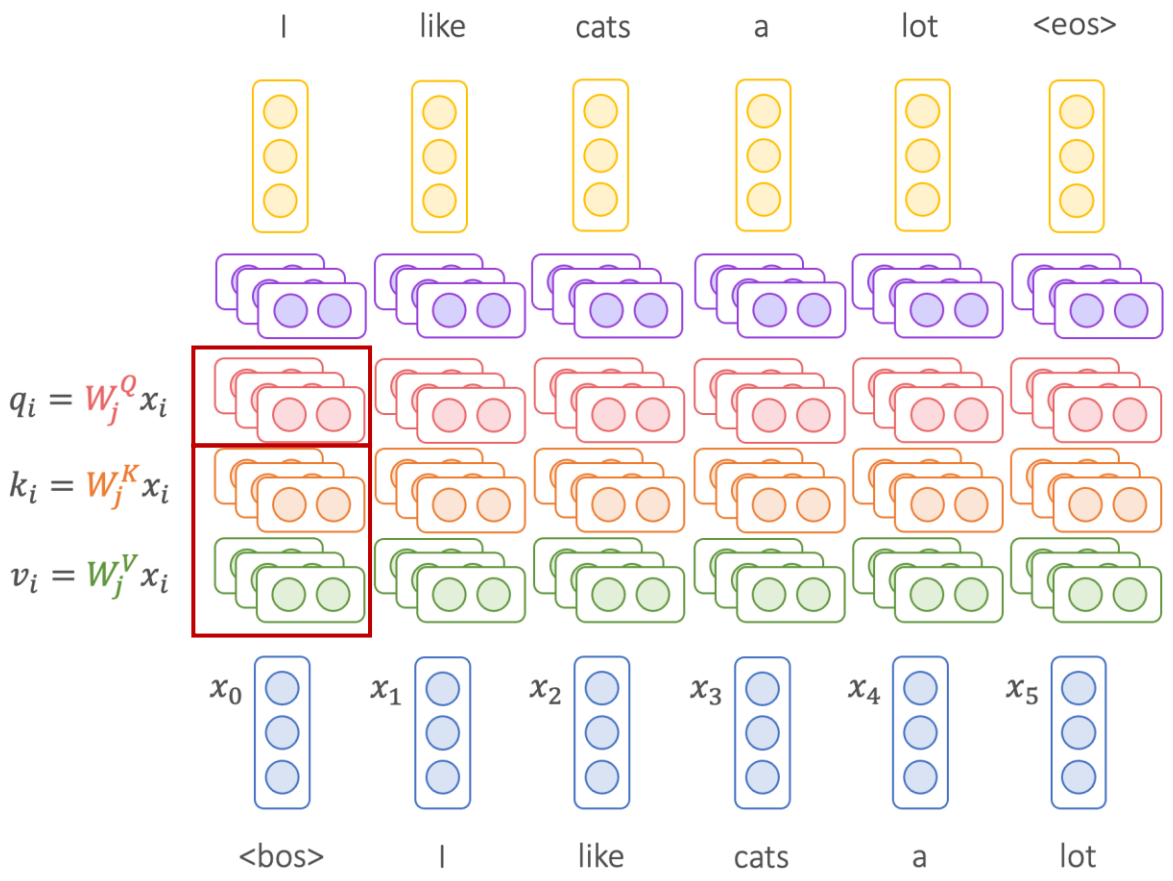
Transformer Decoder



Transformer Encoder vs. Transformer Decoder

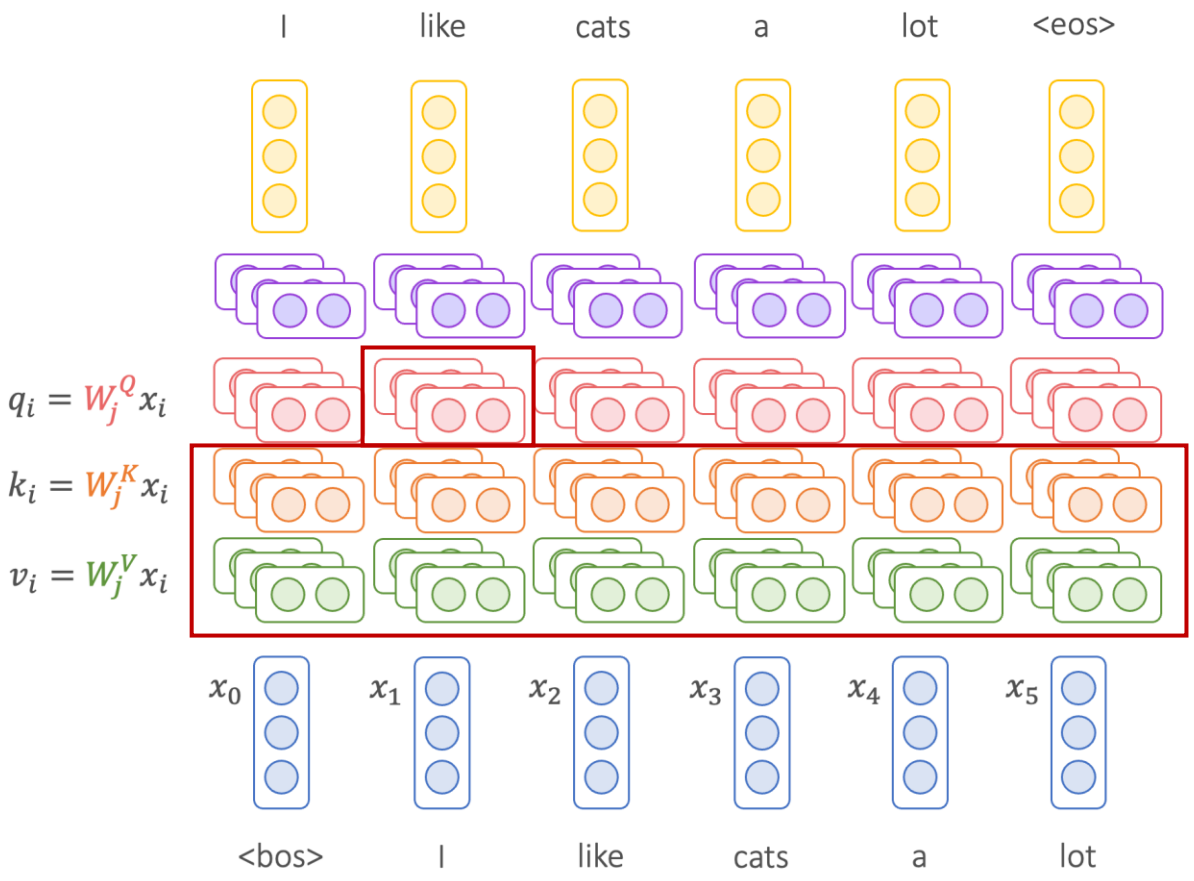


Transformer Encoder

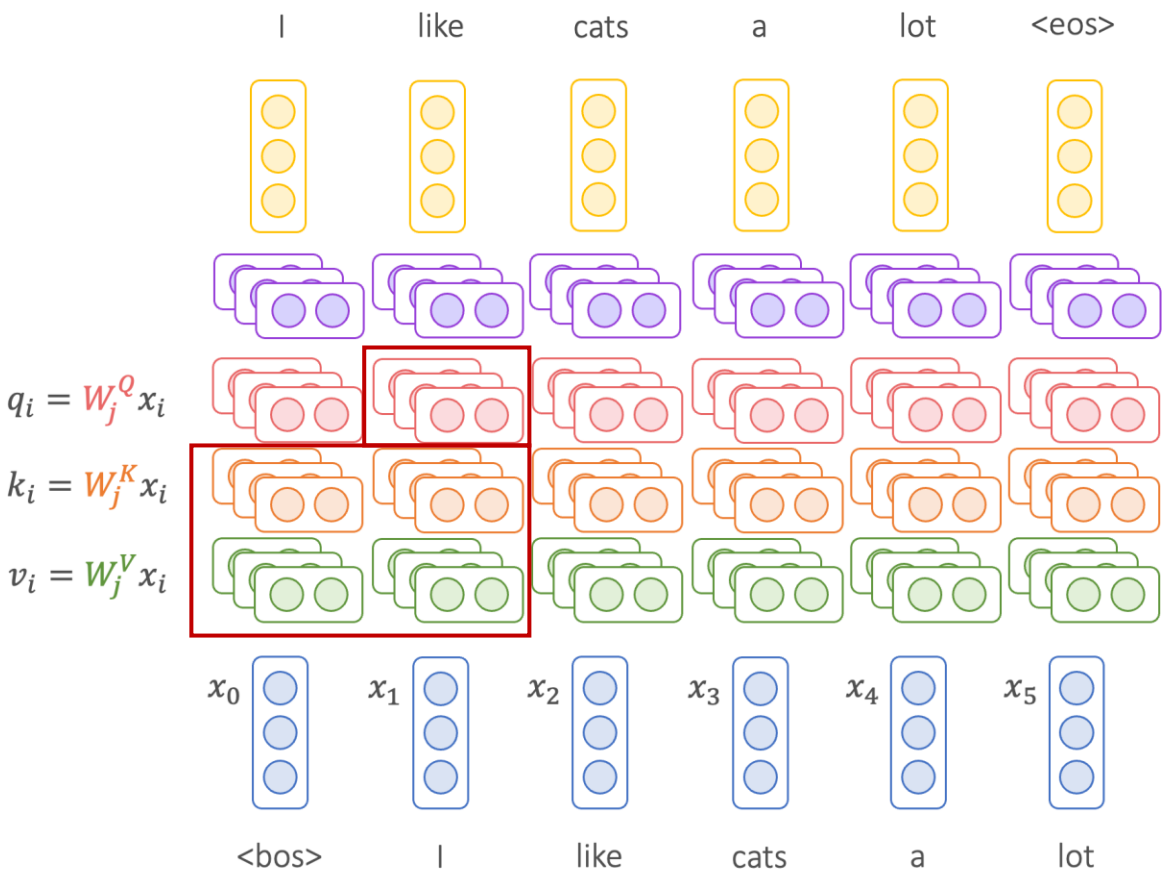


Transformer Decoder

Transformer Encoder vs. Transformer Decoder

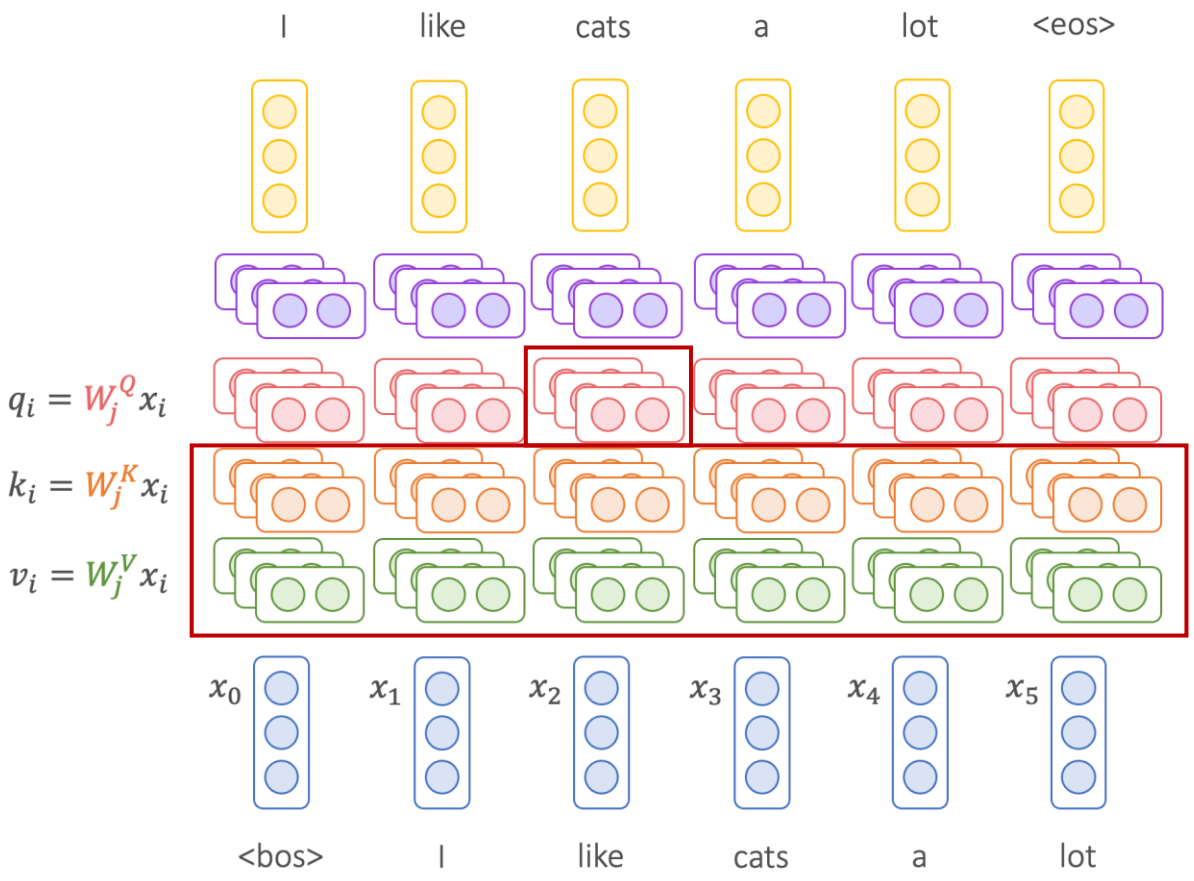


Transformer Encoder

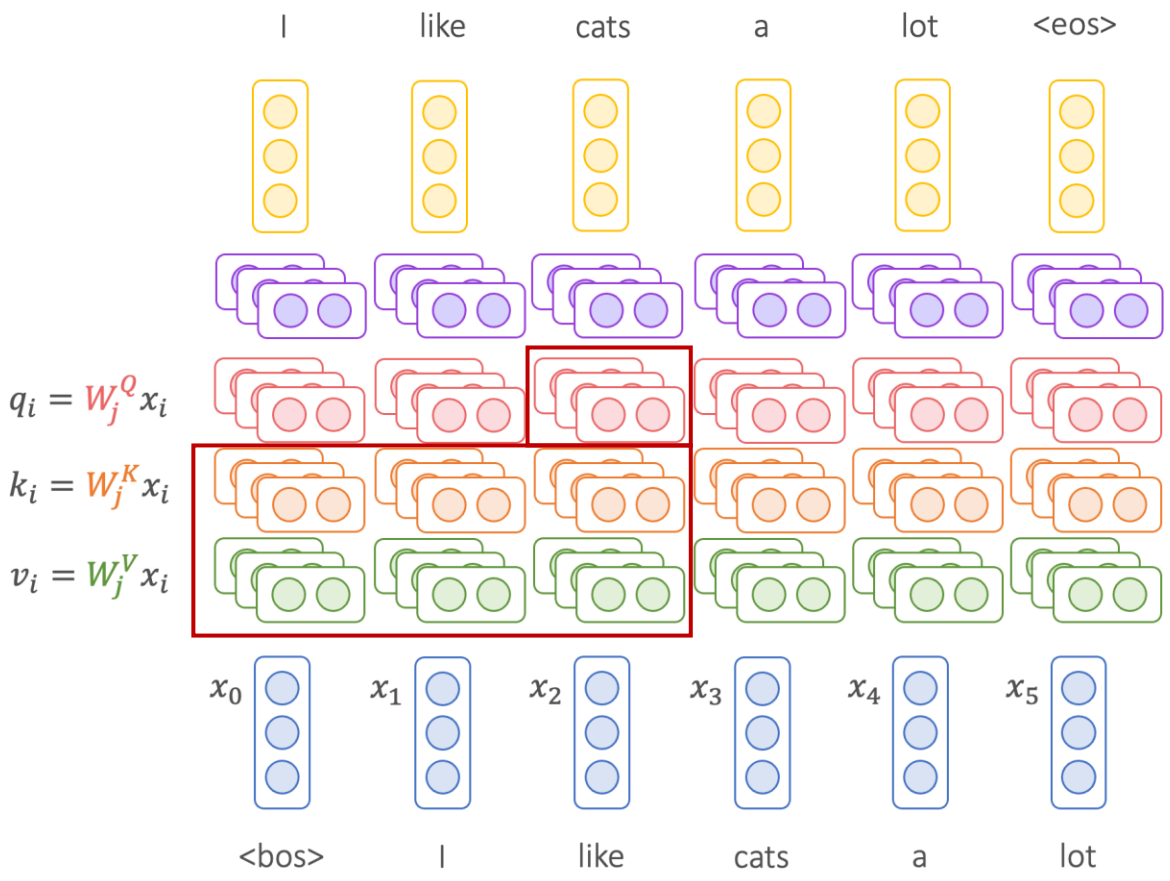


Transformer Decoder

Transformer Encoder vs. Transformer Decoder



Transformer Encoder

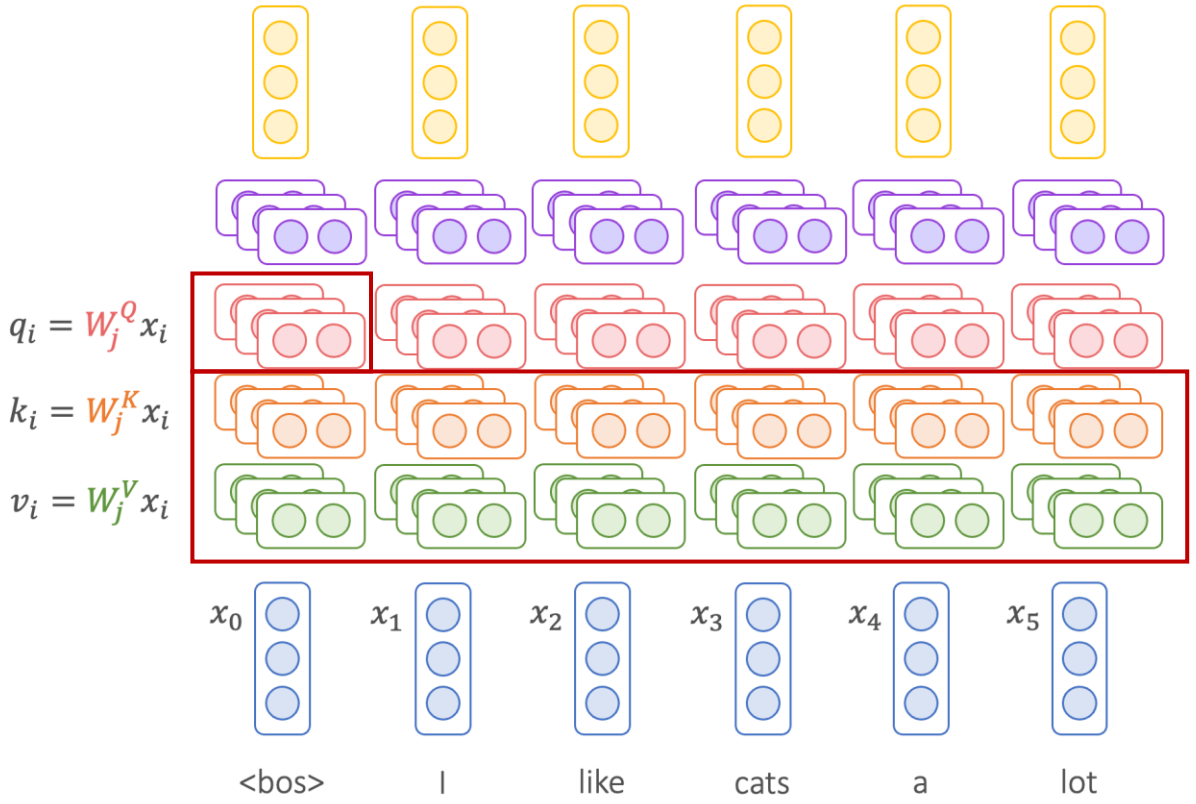


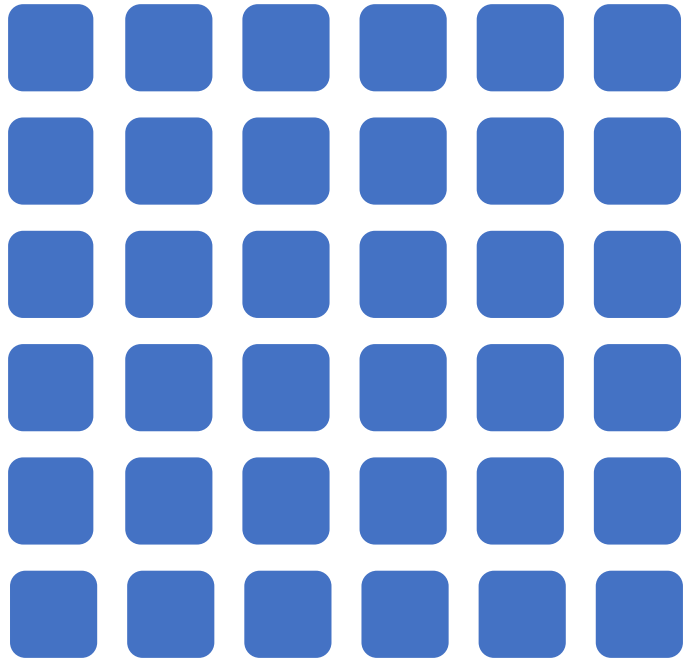
Transformer Decoder

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

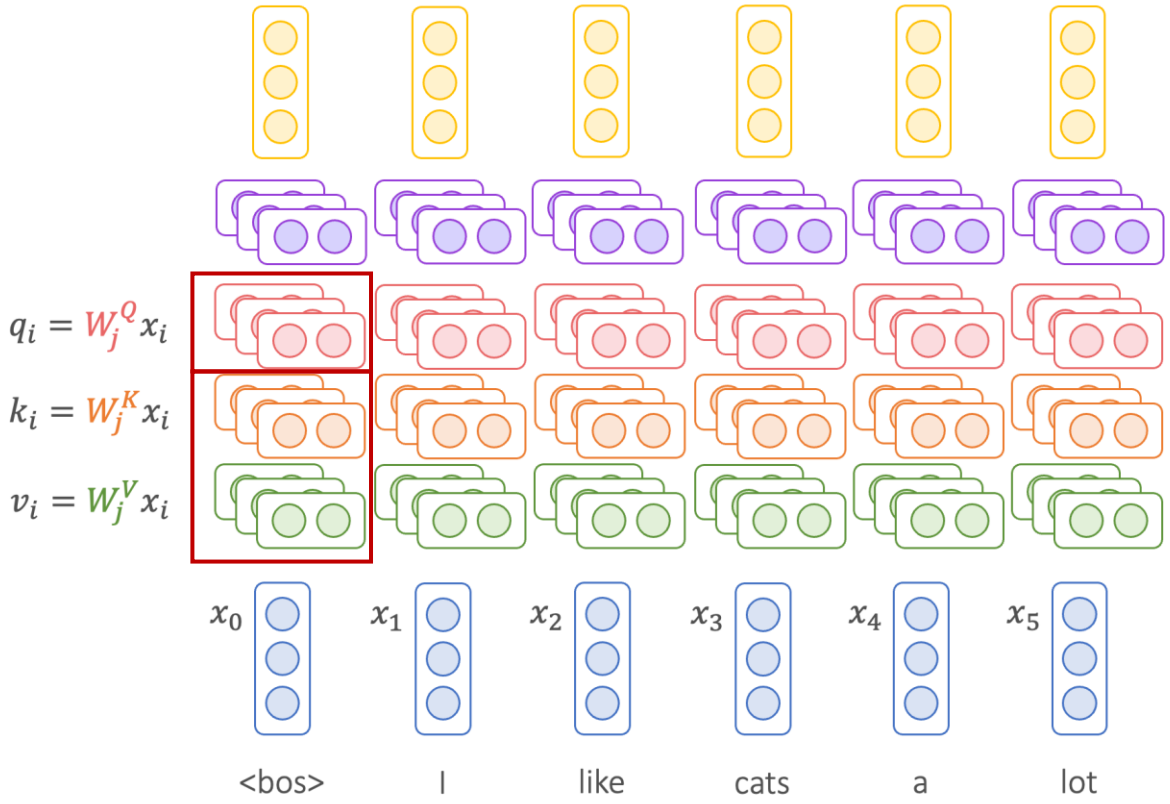
Masked Attention for Transformer Encoder

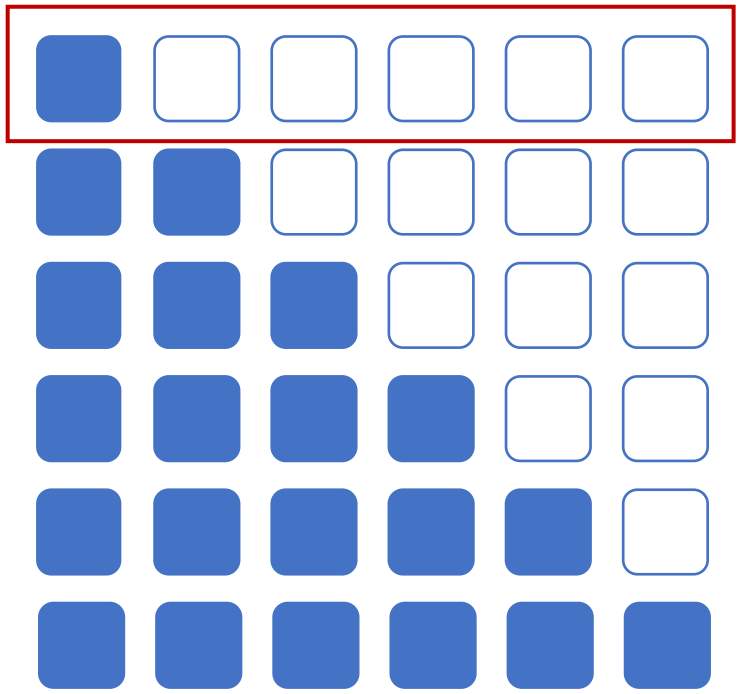




No Masking

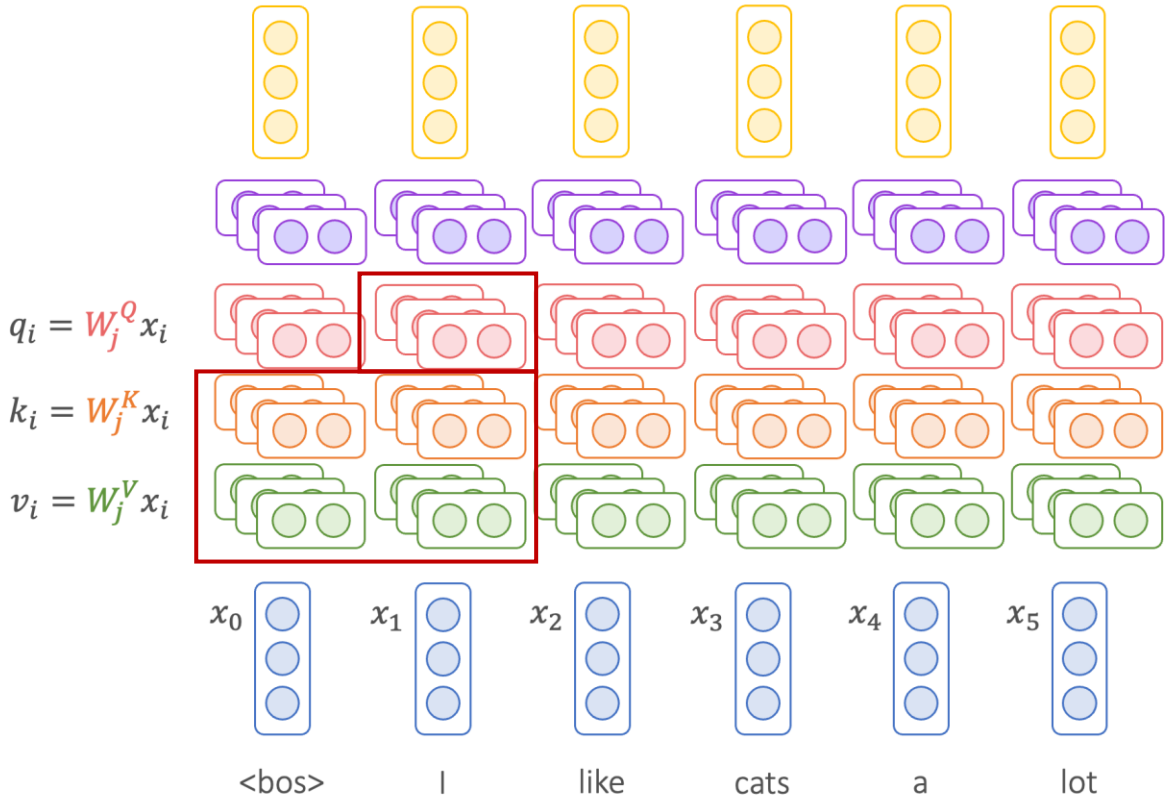
Masked Attention for Transformer Decoder

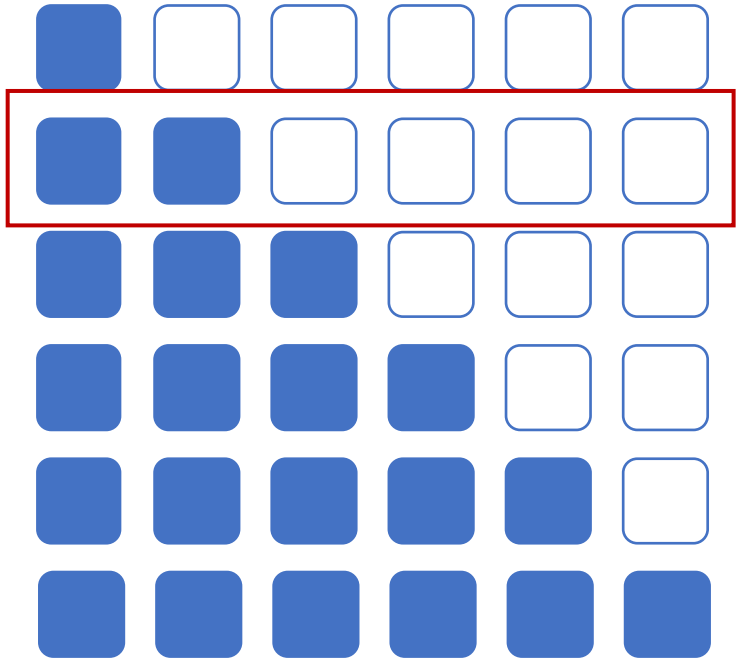




Causal Masking

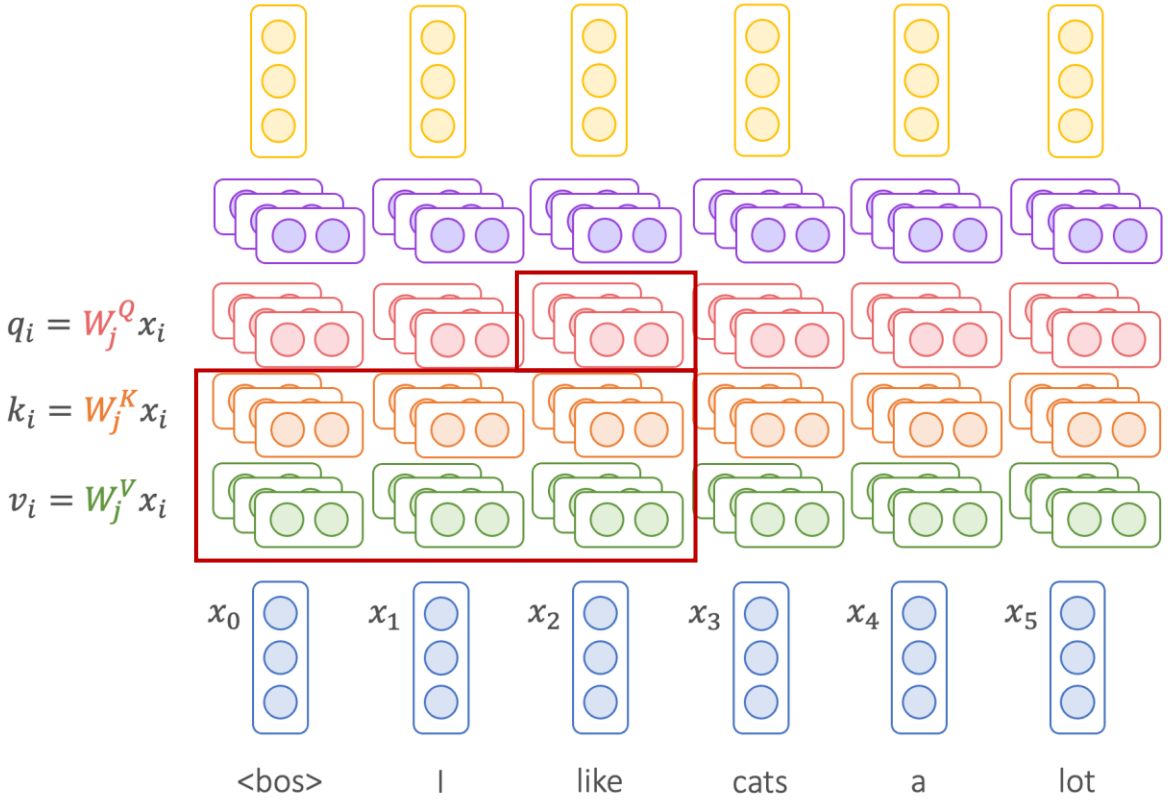
Masked Attention for Transformer Decoder

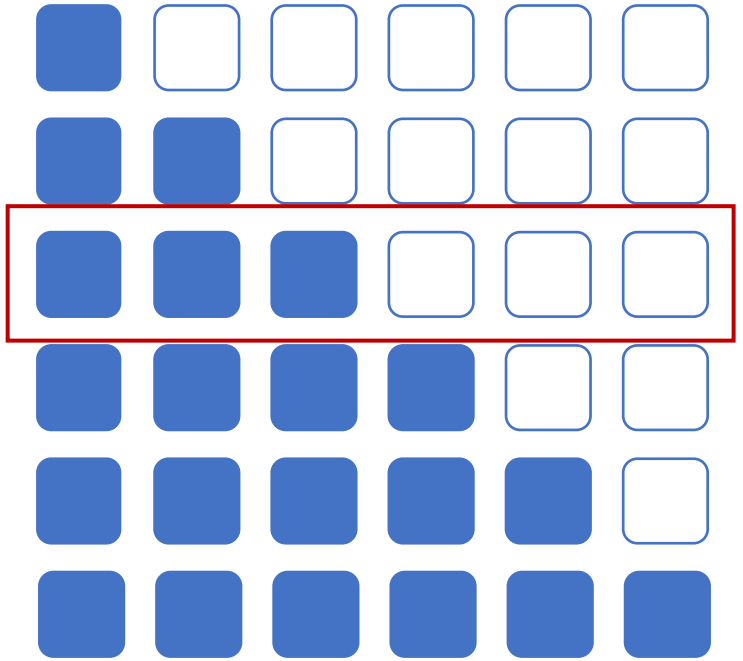




Causal Masking

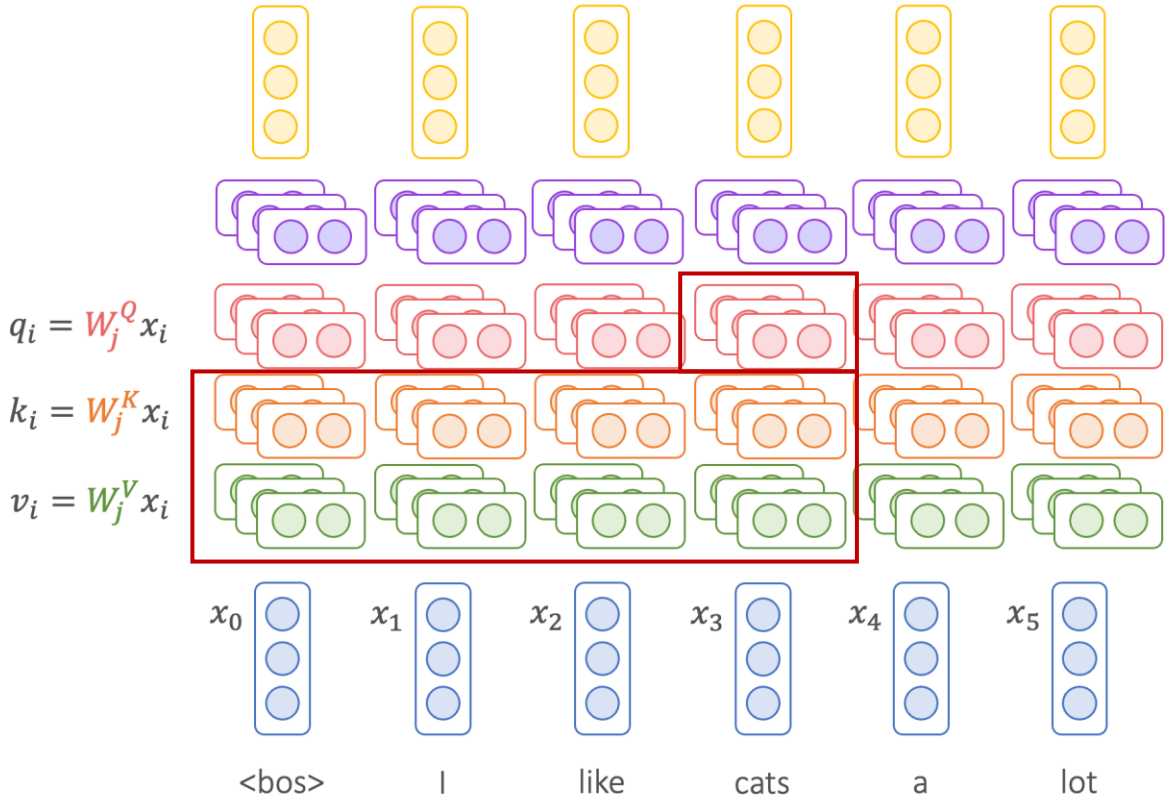
Masked Attention for Transformer Decoder

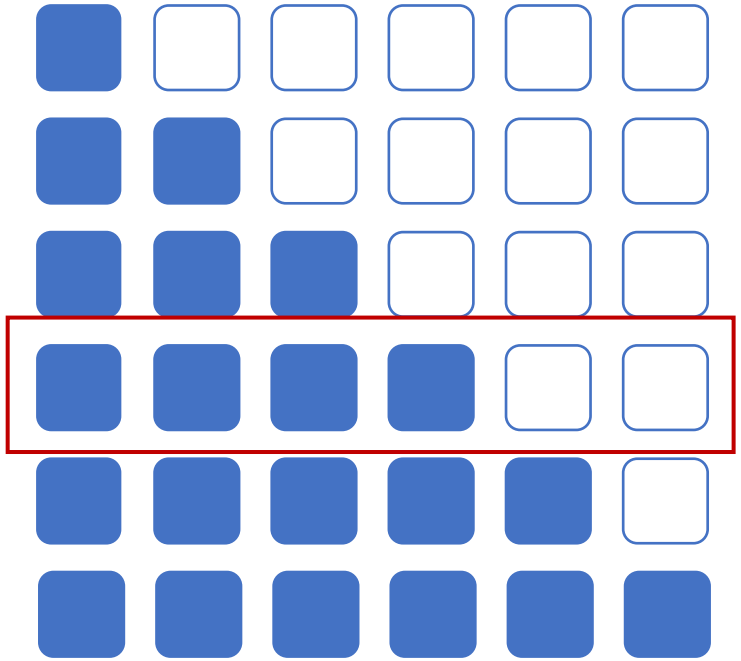




Causal Masking

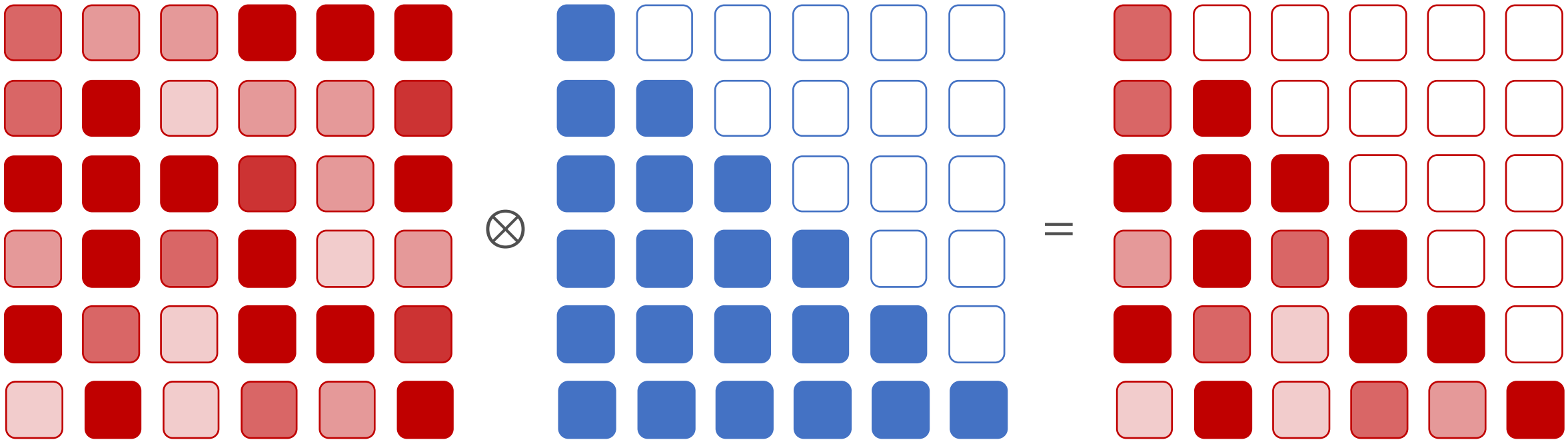
Masked Attention for Transformer Decoder





Causal Masking

Masked Attention: Implementation

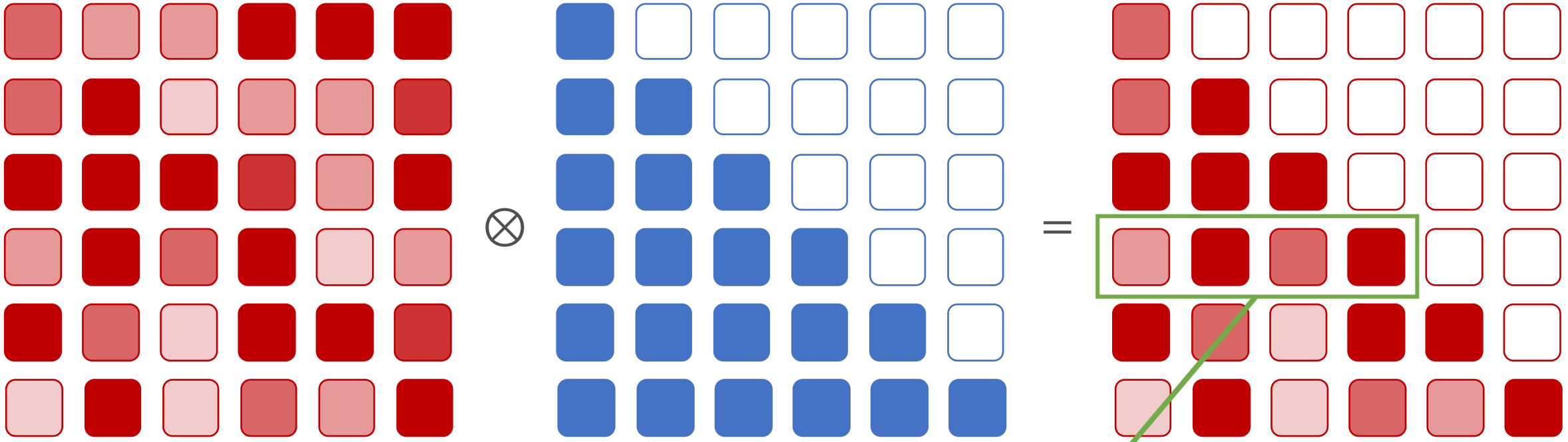


All-Pair Attention Scores

Causal Masking

Causal Attention Scores

Masked Attention: Implementation



All-Pair Attention Scores

Causal Masking

Causal Attention Scores

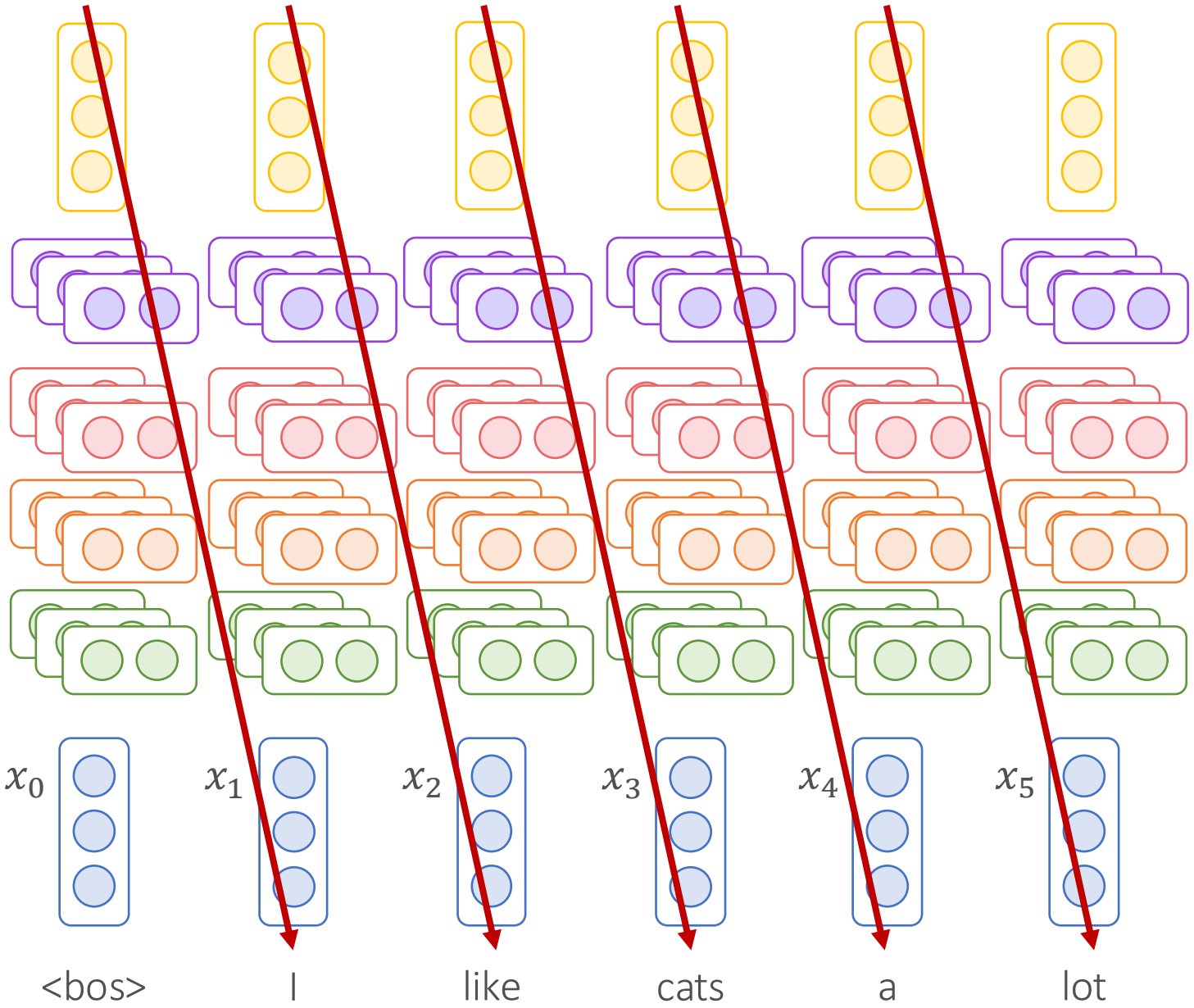
Normalize attention weights
& Weighted average value vectors

Transformer Decoder

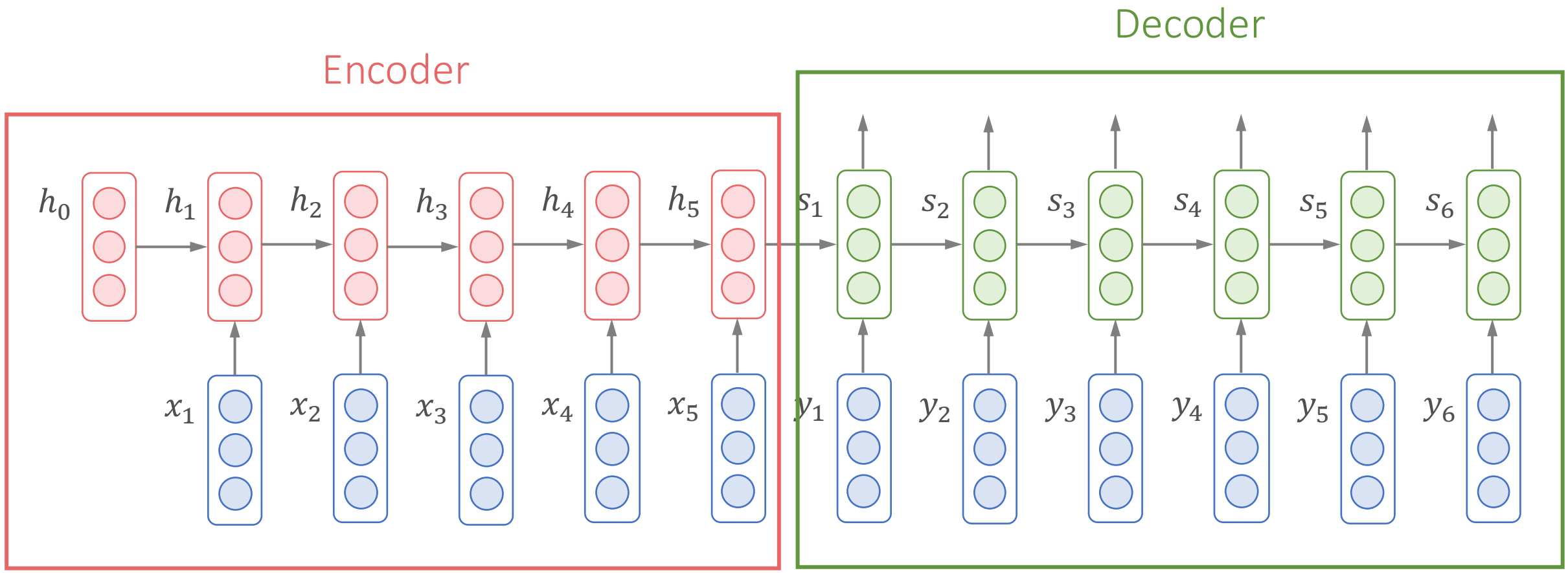
$q_i = W_j^Q x_i$

$k_i = W_j^K x_i$

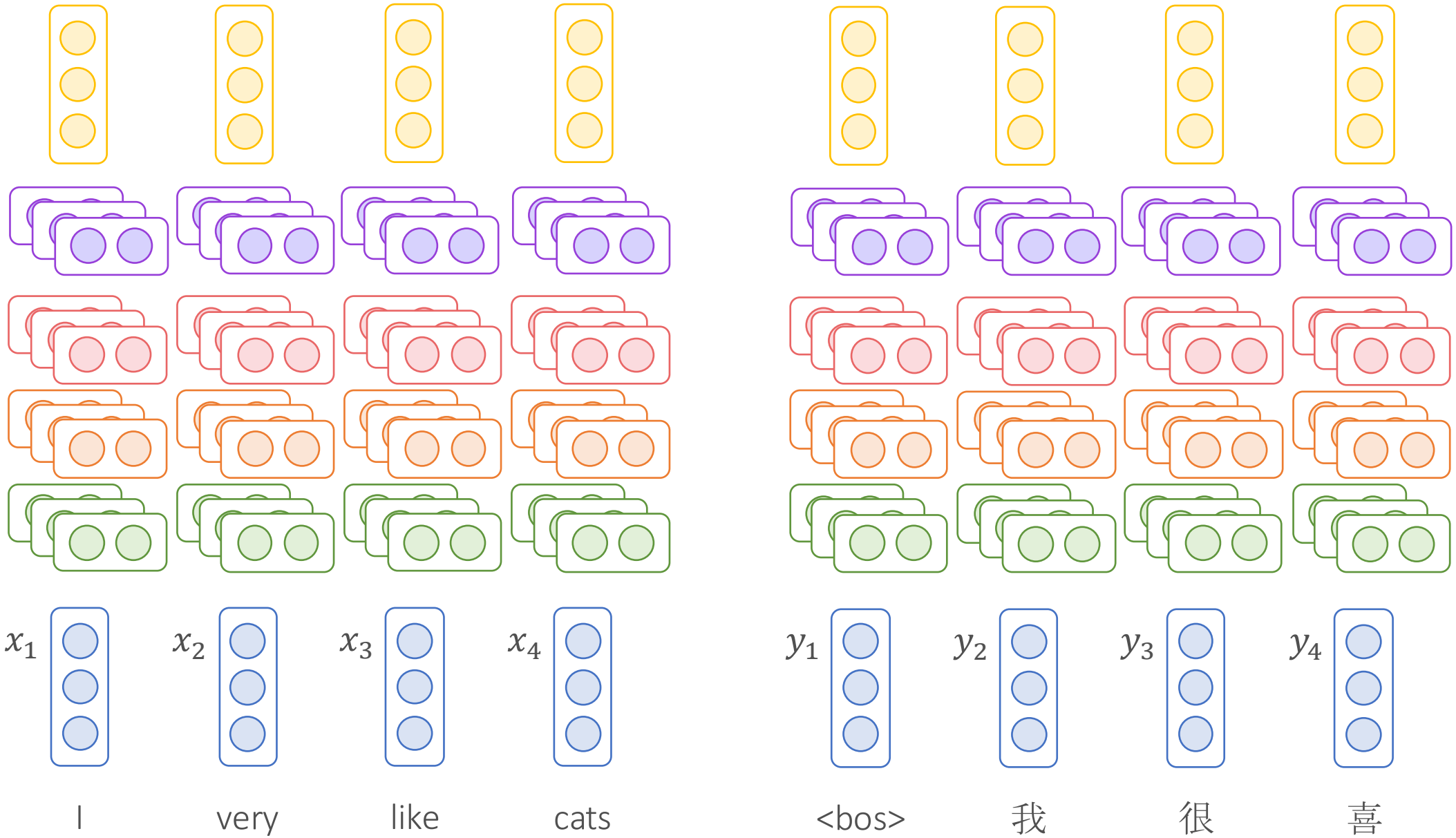
$v_i = W_j^V x_i$



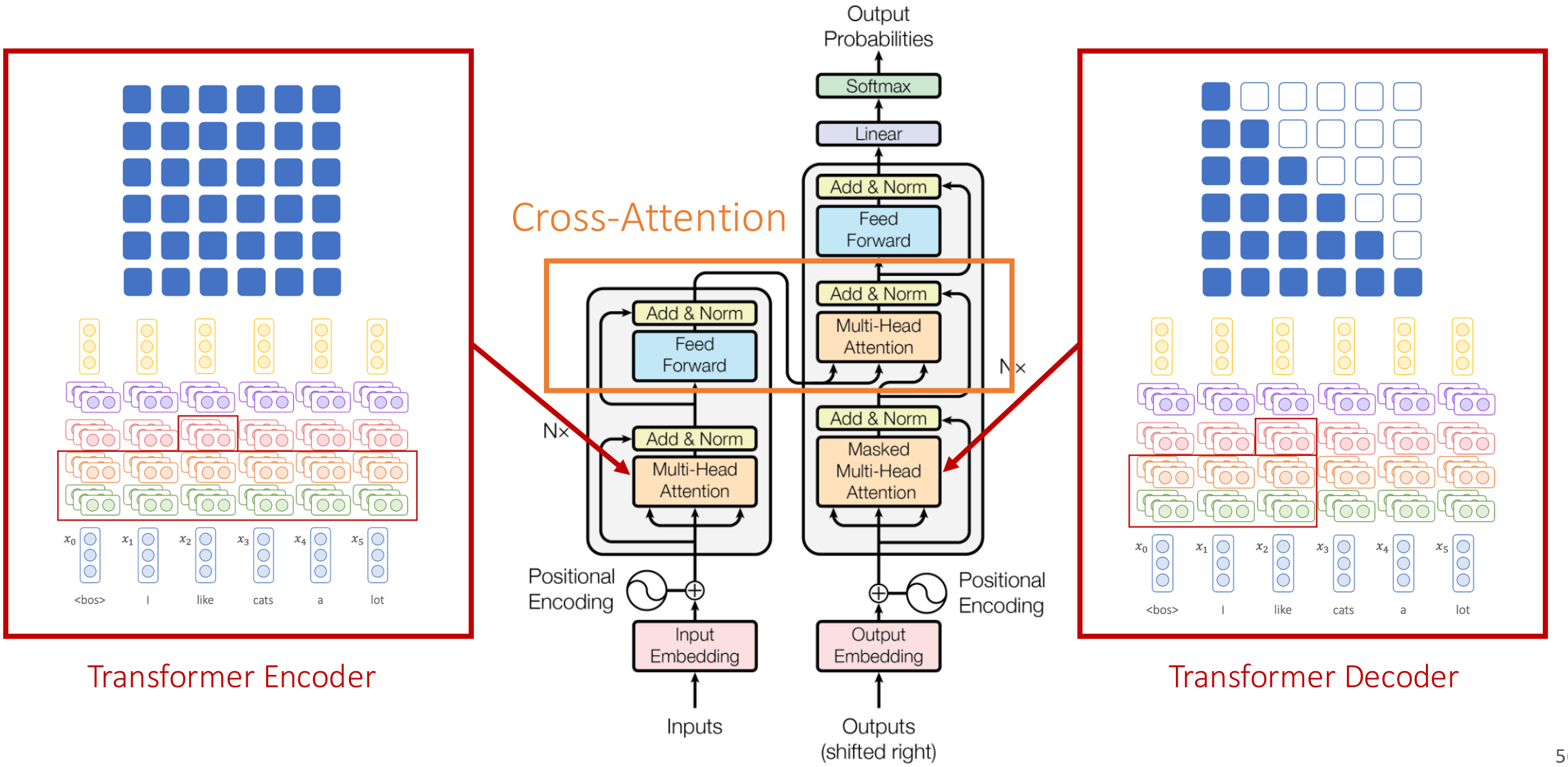
How About Encoder-Decoder (Sequence-to-Sequence)?



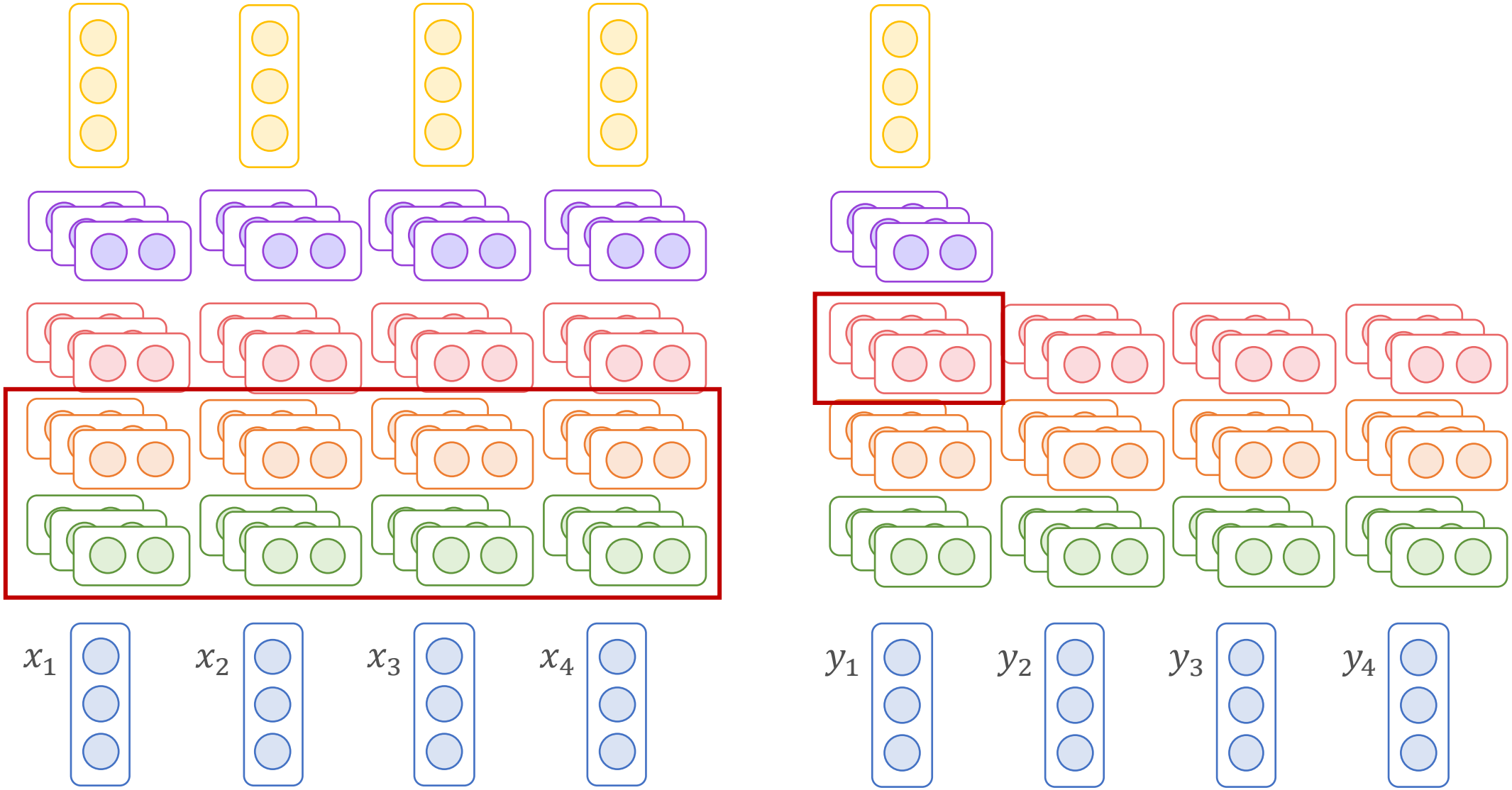
Transformer Encoder-Decoder (Sequence-to-Sequence)



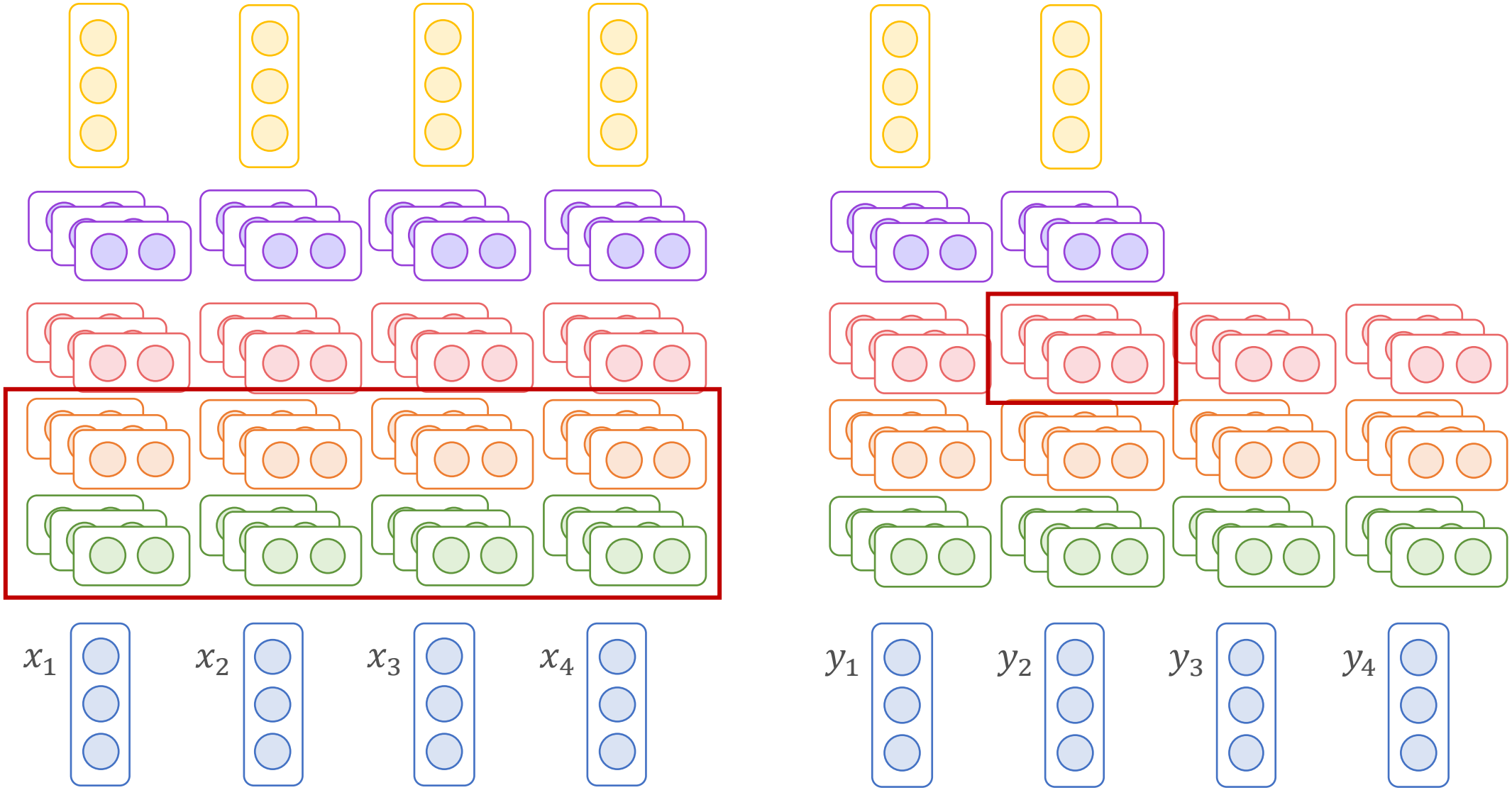
Transformer Encoder-Decoder (Sequence-to-Sequence)



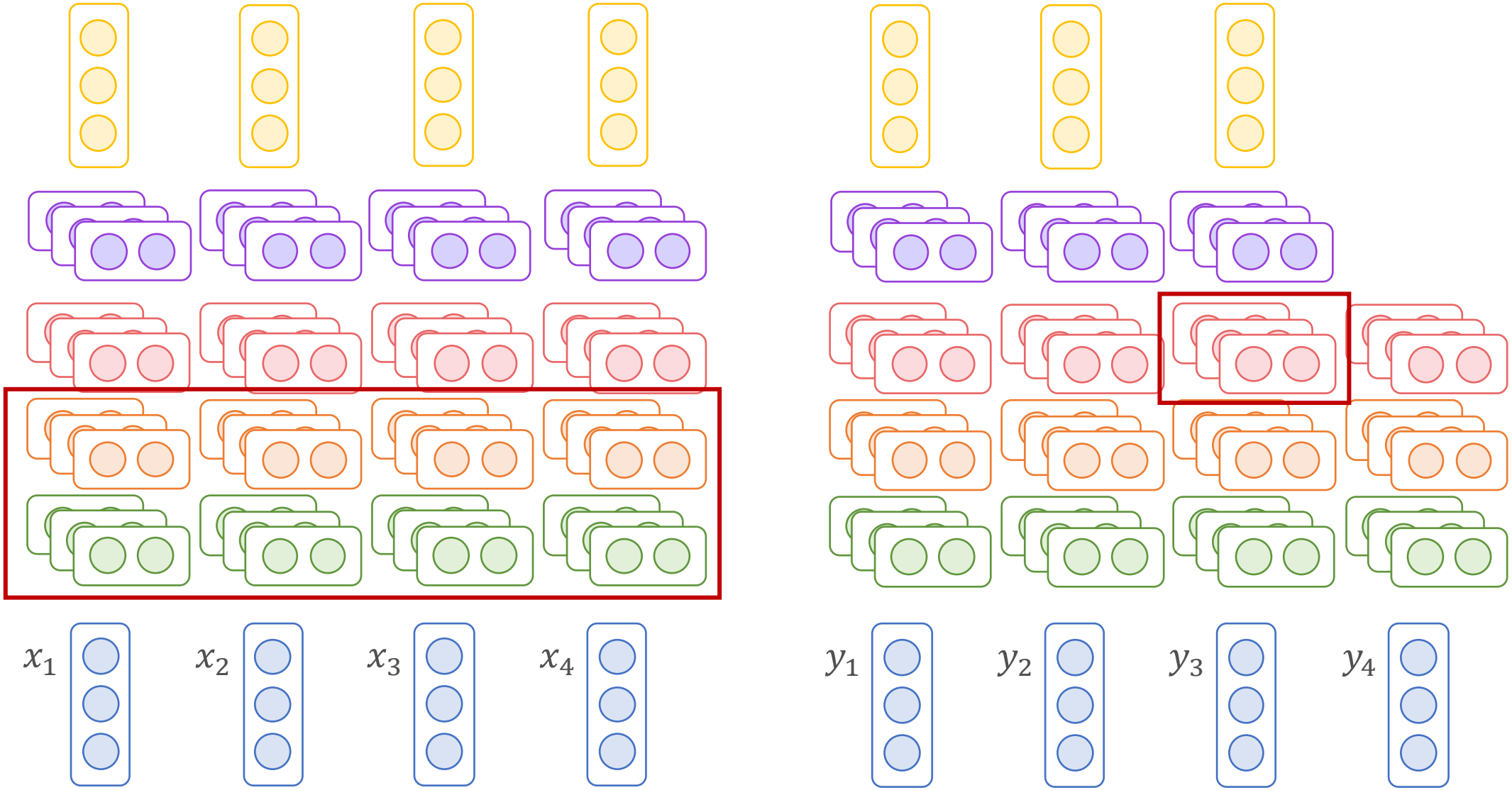
Cross-Attention



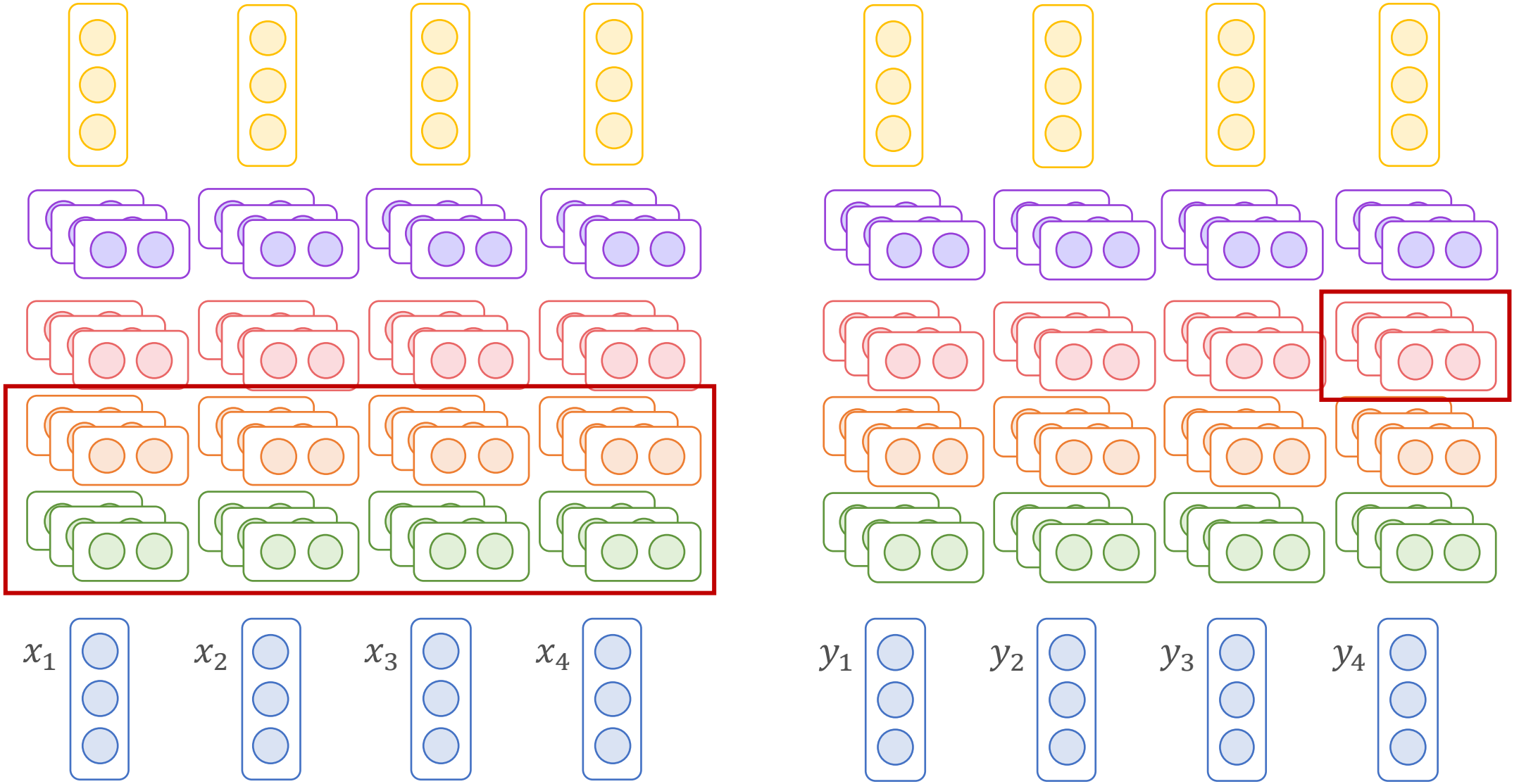
Cross-Attention



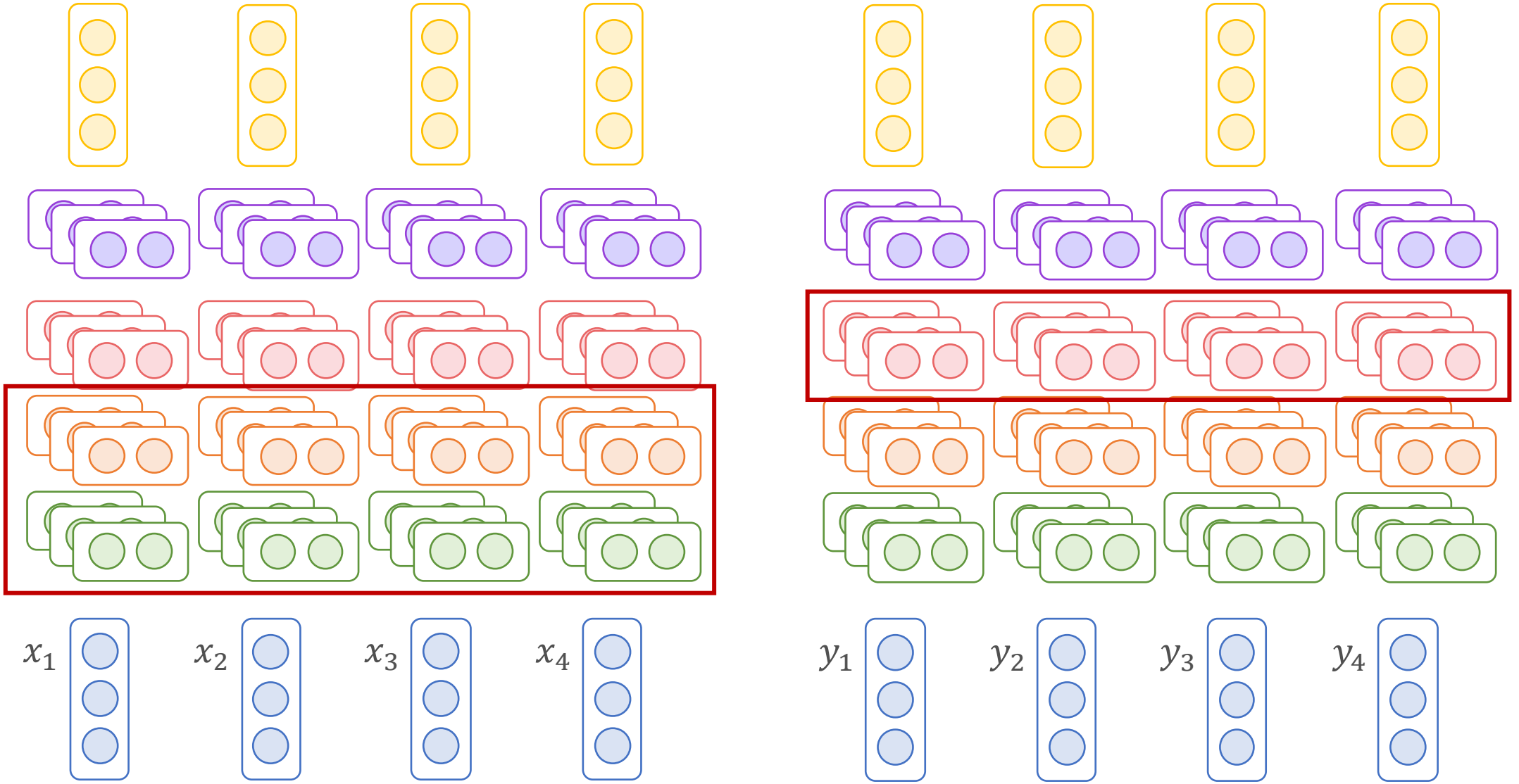
Cross-Attention



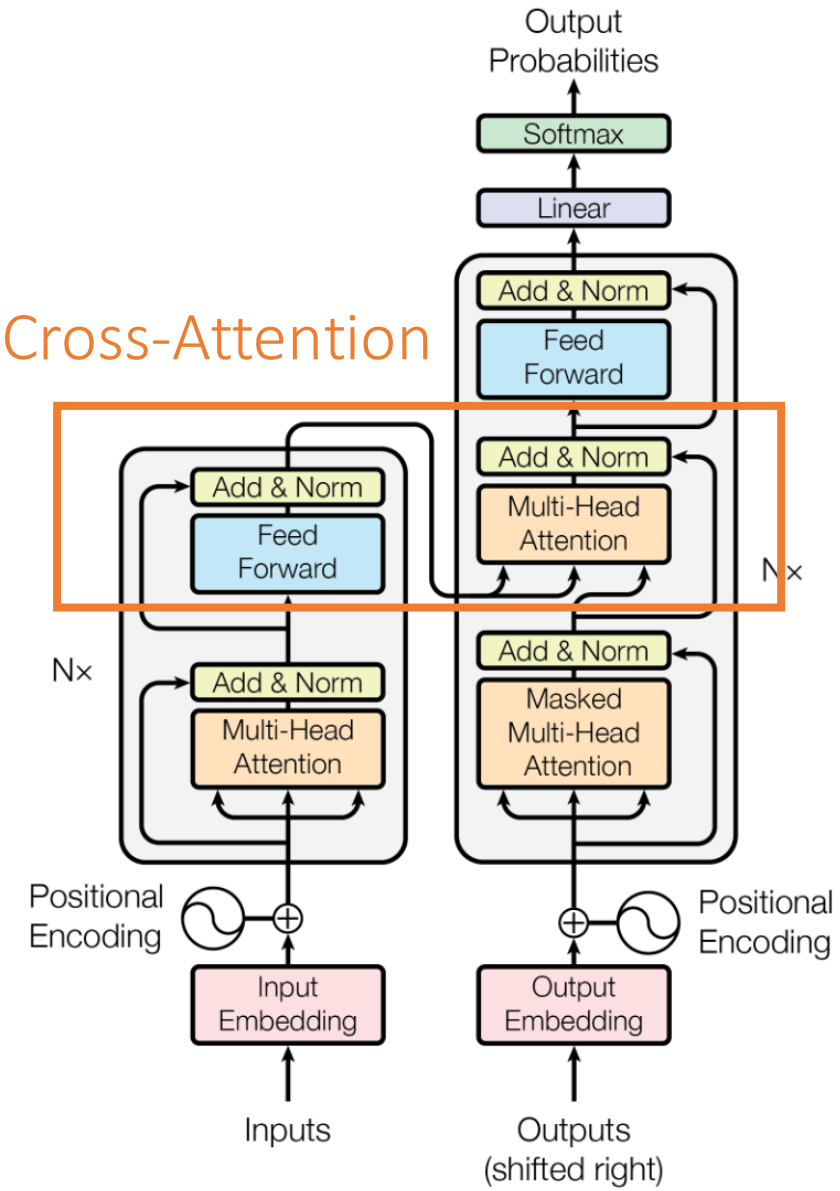
Cross-Attention



Cross-Attention



Transformer



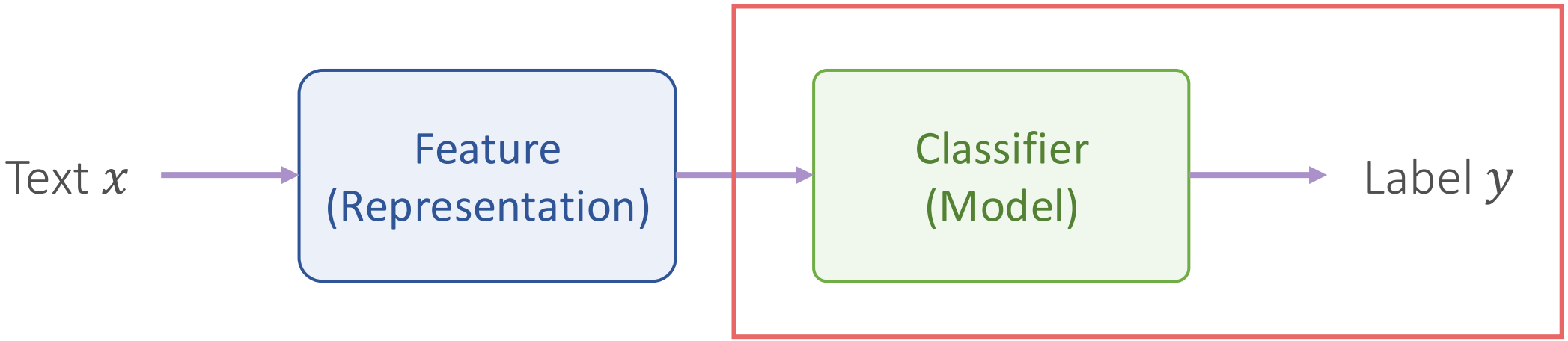
Cross-Attention

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

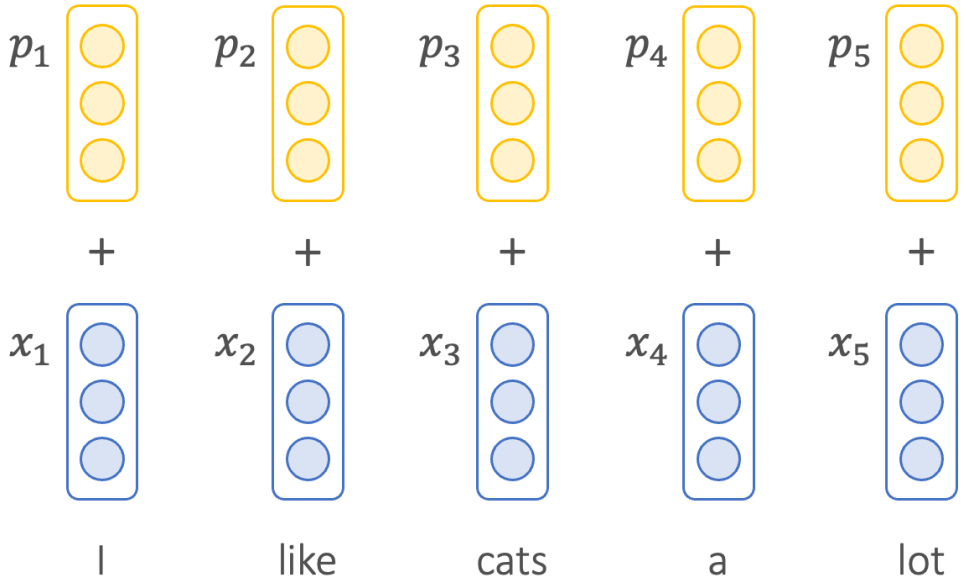


- Teach the model how to make prediction y
- Logistic regression, neural networks, CNN, RNN, LSTM, Transformers

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	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$

Absolute Positional Encoding

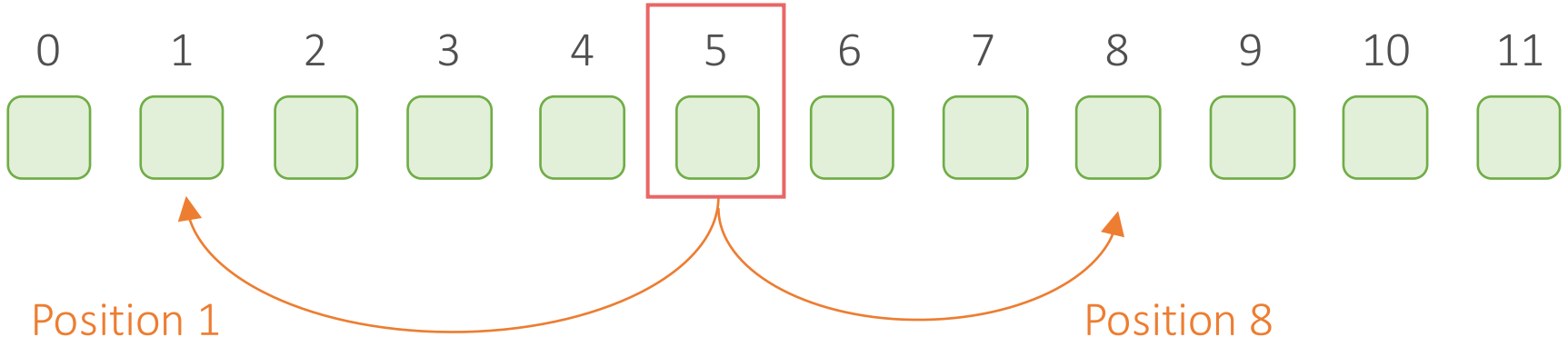
$$x_i \leftarrow x_i + PE_i$$

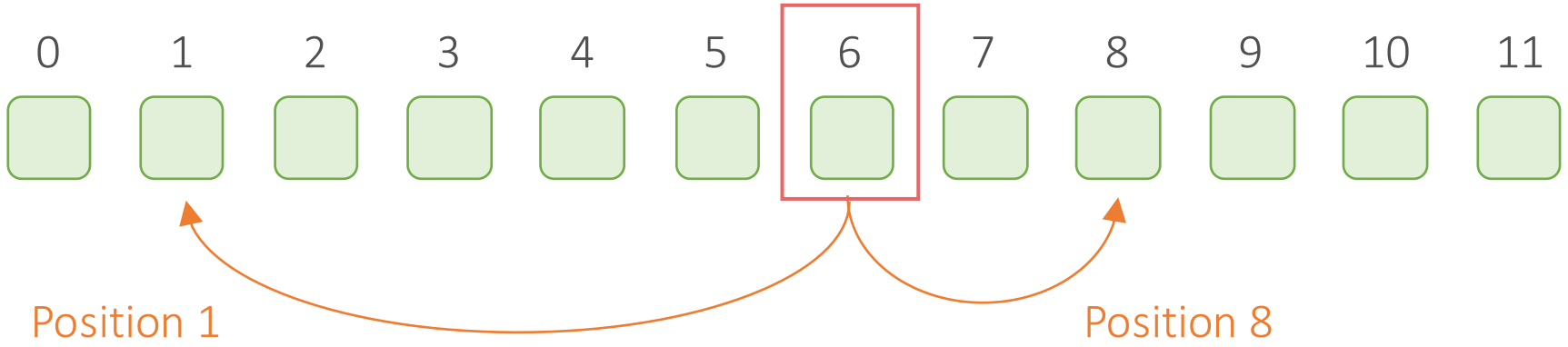


$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

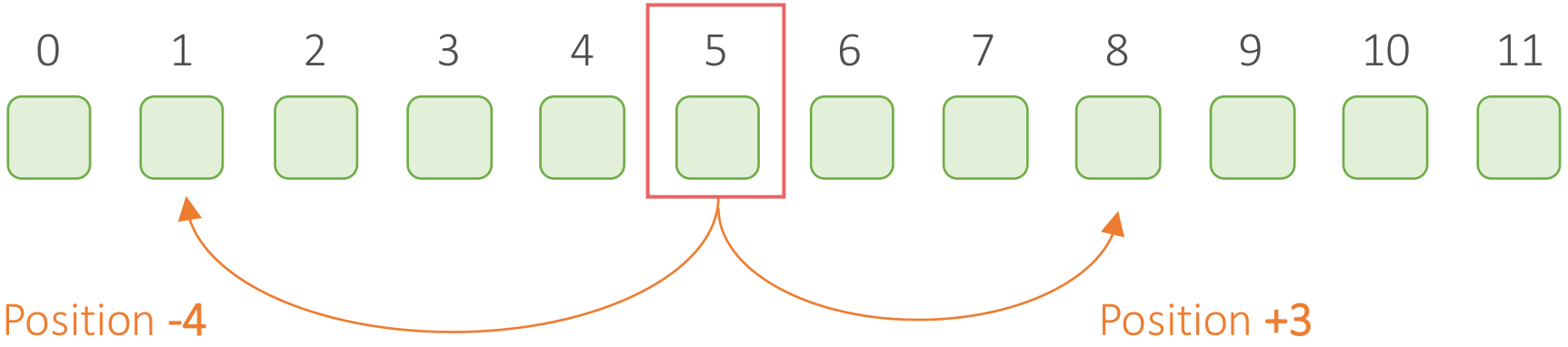
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

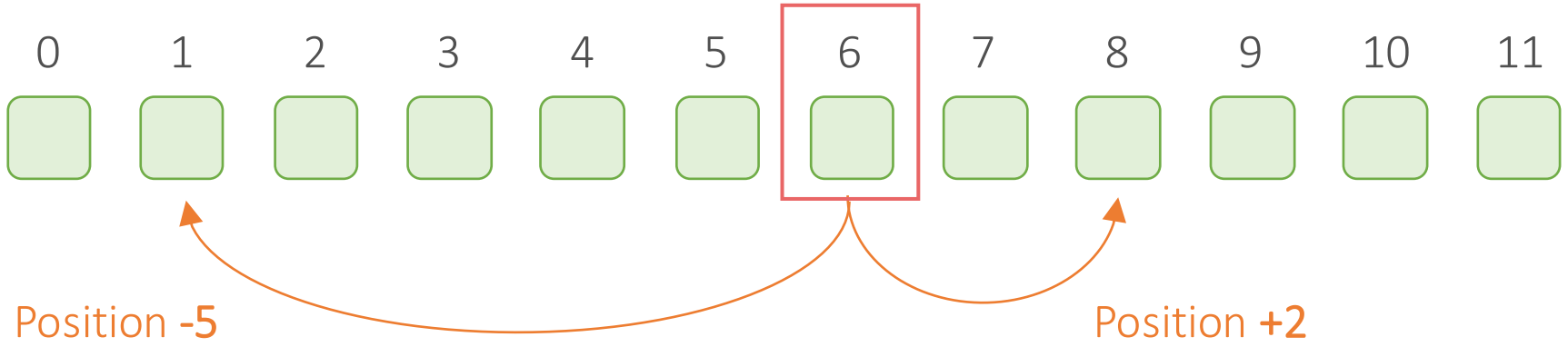
Absolute Position





Relative Position

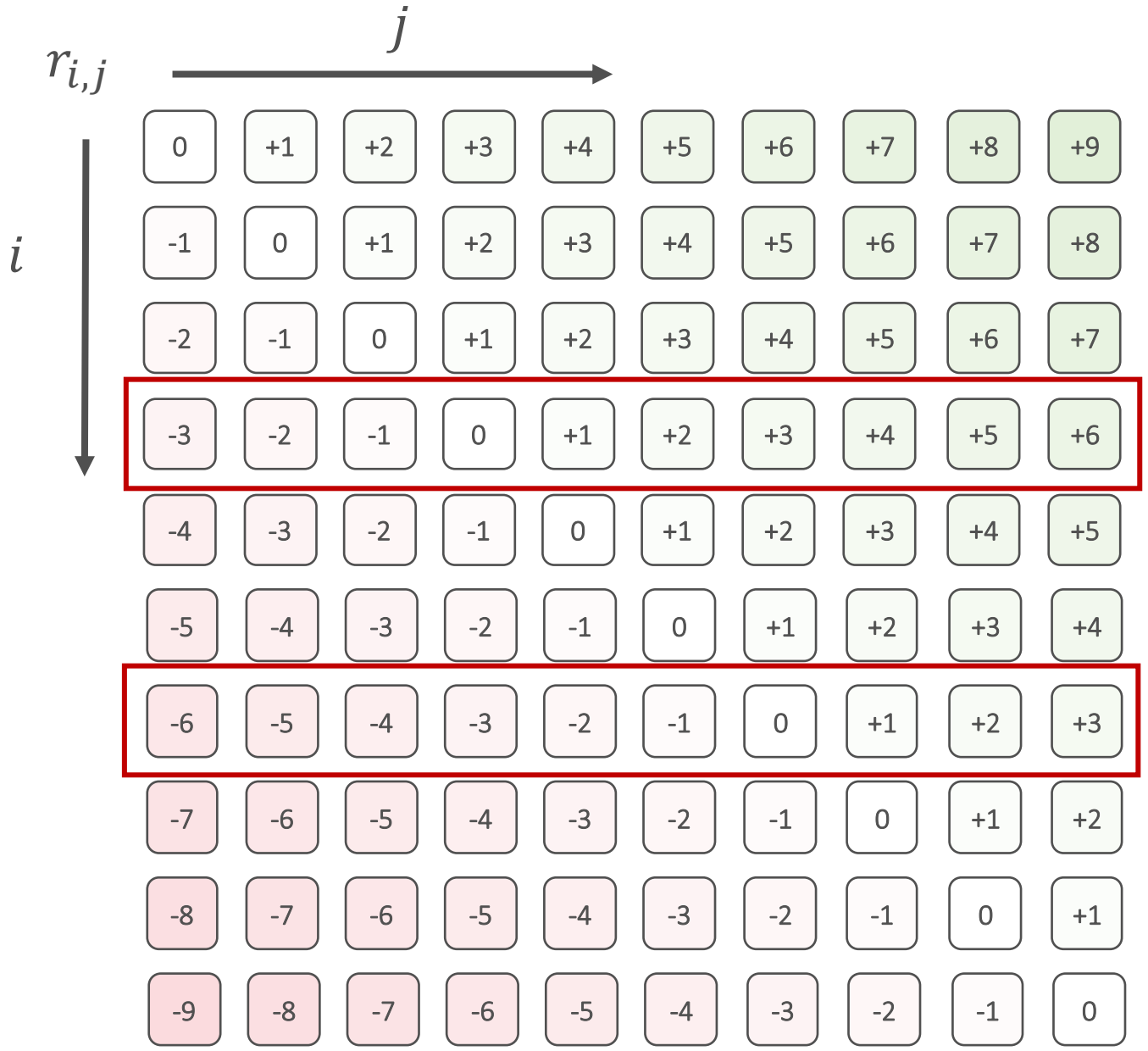




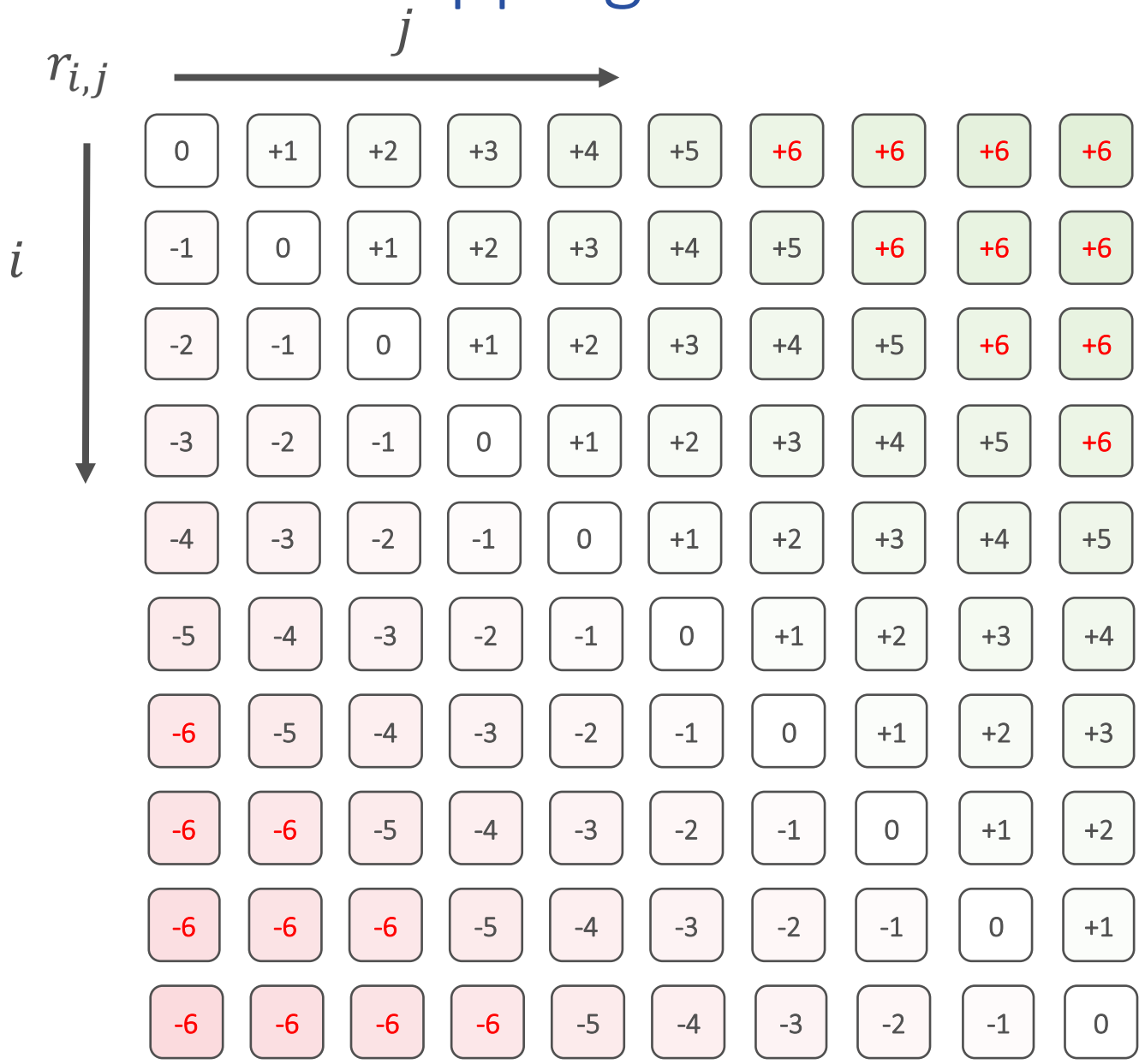
Why Relative Position?

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

Relative Position

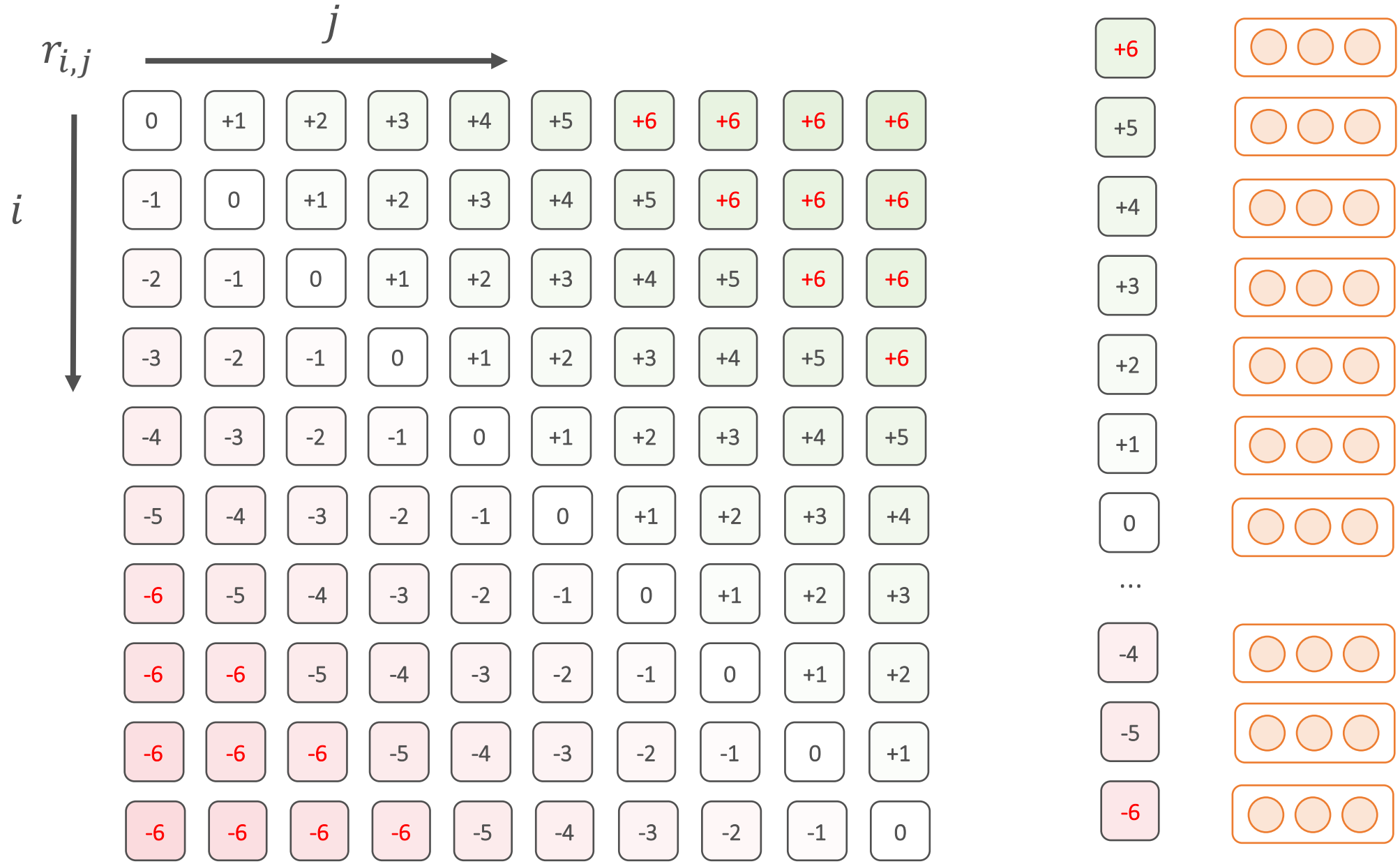


Relative Position with Clipping



Limited relative positions

Map Relative Positions to Embeddings



Self-Attention

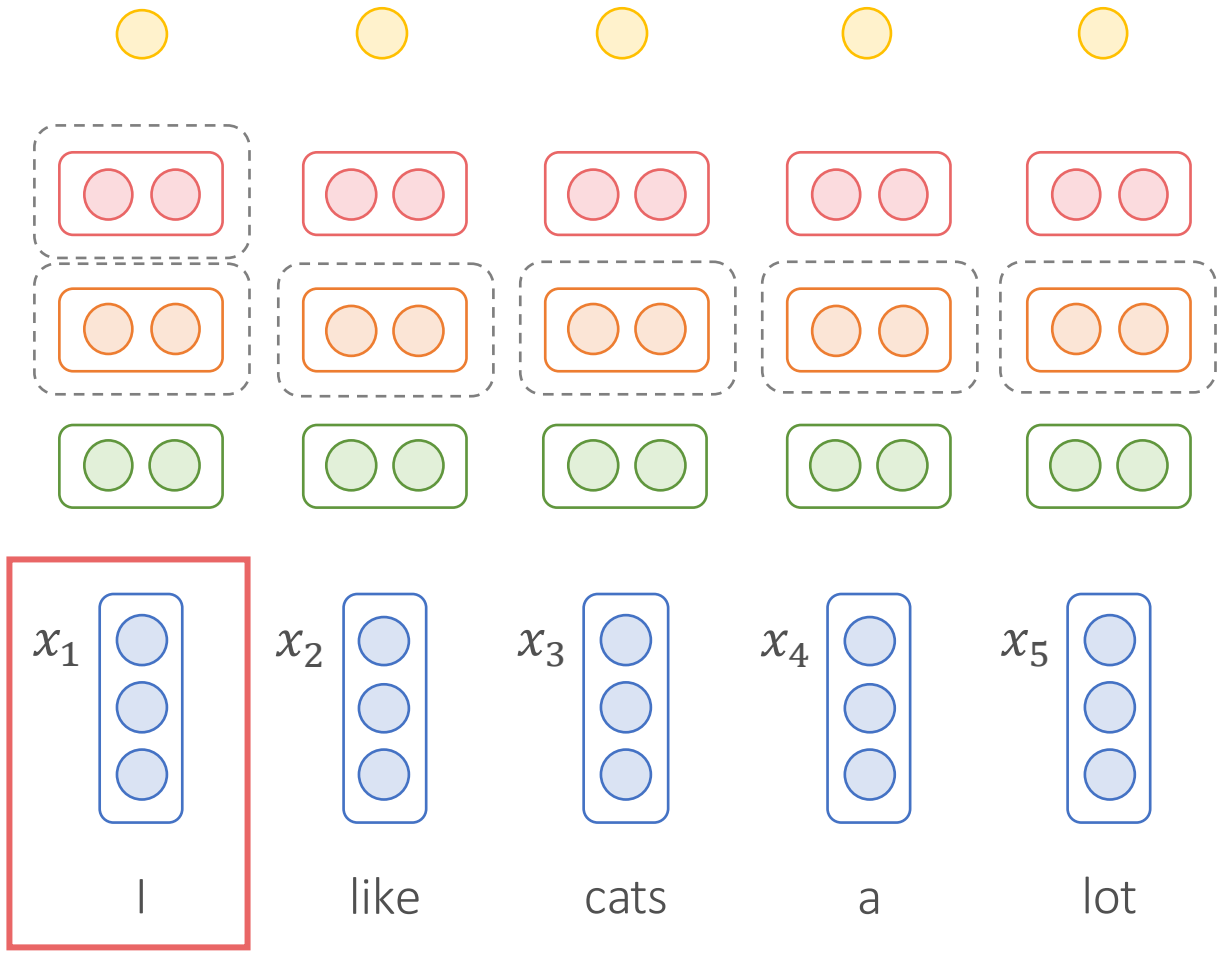
$$\alpha_{1,i} = \text{softmax}\left(\frac{q_1 \cdot k_i}{\sqrt{d}}\right)$$

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

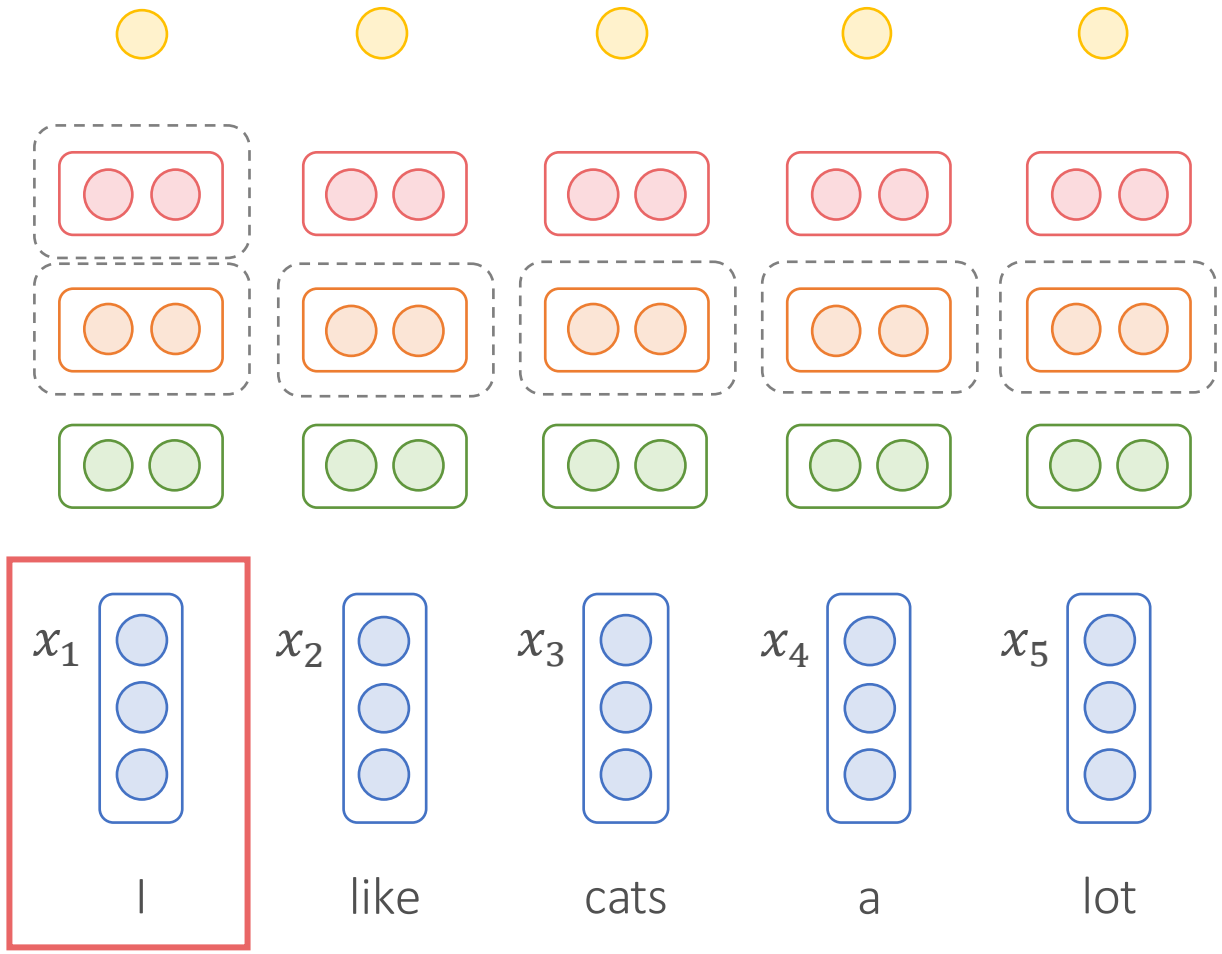
$$\alpha_{1,i} = \text{softmax}\left(\frac{W^Q x_1 \cdot W^K x_i}{\sqrt{d}}\right)$$

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

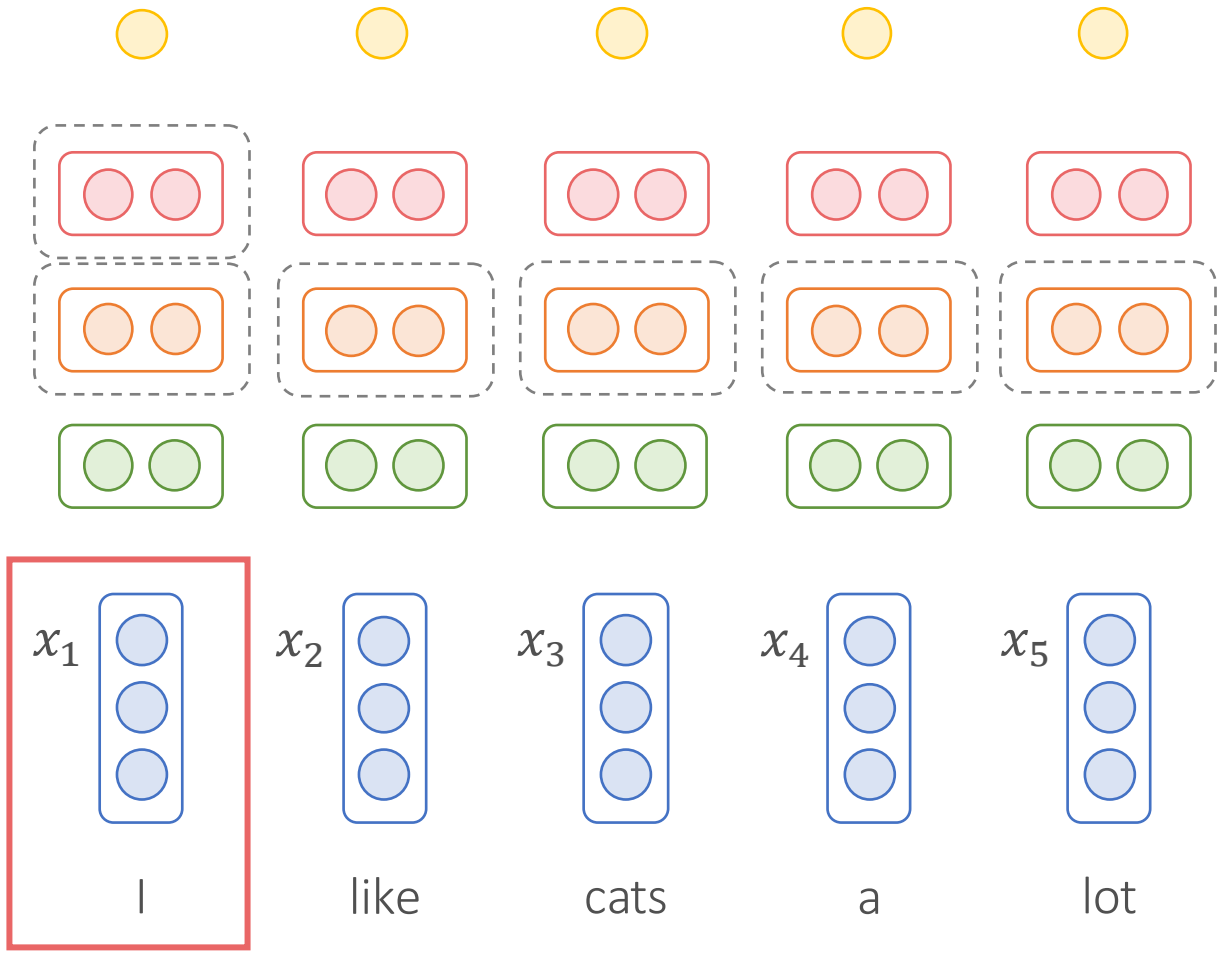
$$\alpha_{1,i} = \text{softmax}\left(\frac{W^Q x_1 \cdot W^K\left(x_i + RE(r_{1,i})\right)}{\sqrt{d}}\right)$$

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

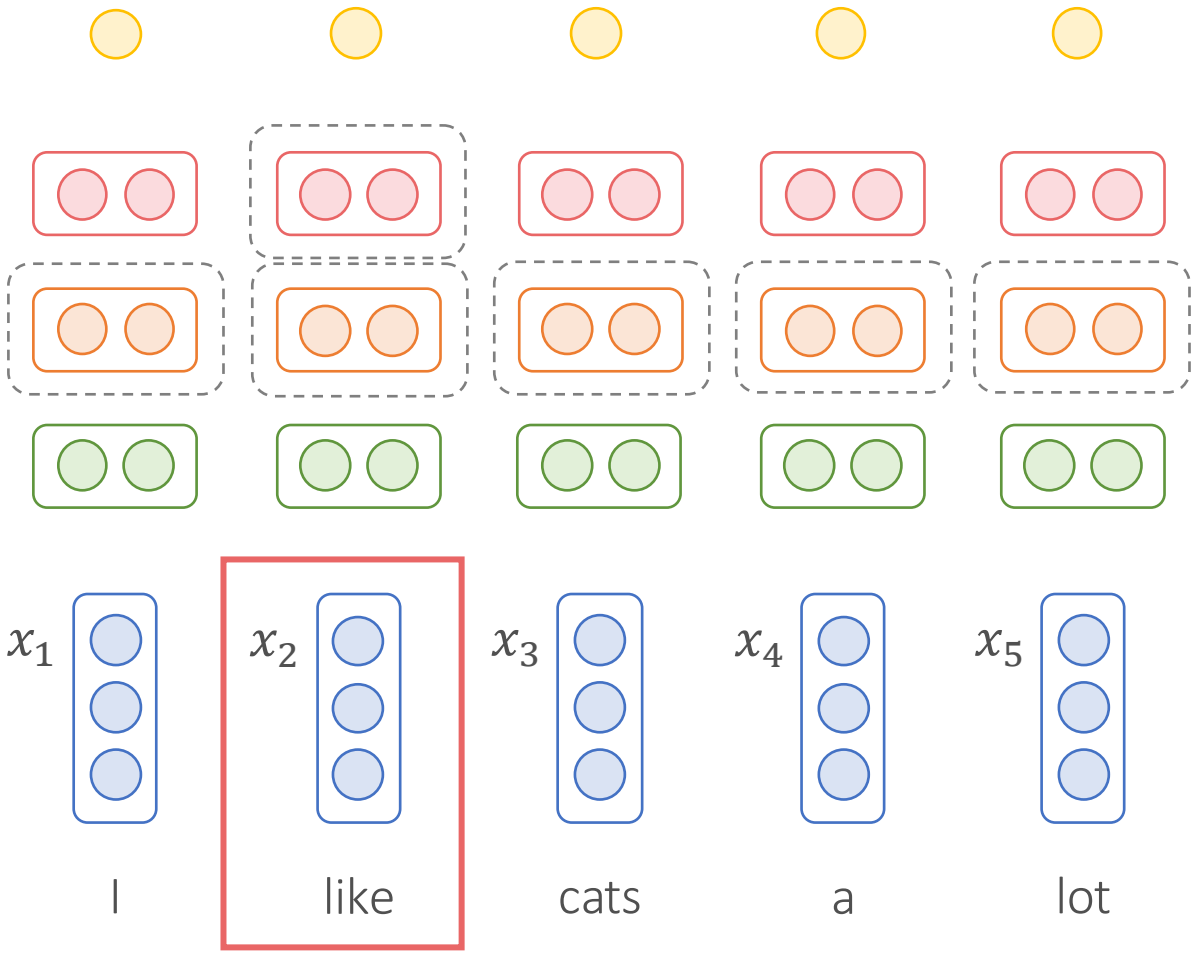
$$\alpha_{2,i} = \text{softmax}\left(\frac{W^Q x_2 \cdot W^K\left(x_i + RE(r_{2,i})\right)}{\sqrt{d}}\right)$$

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

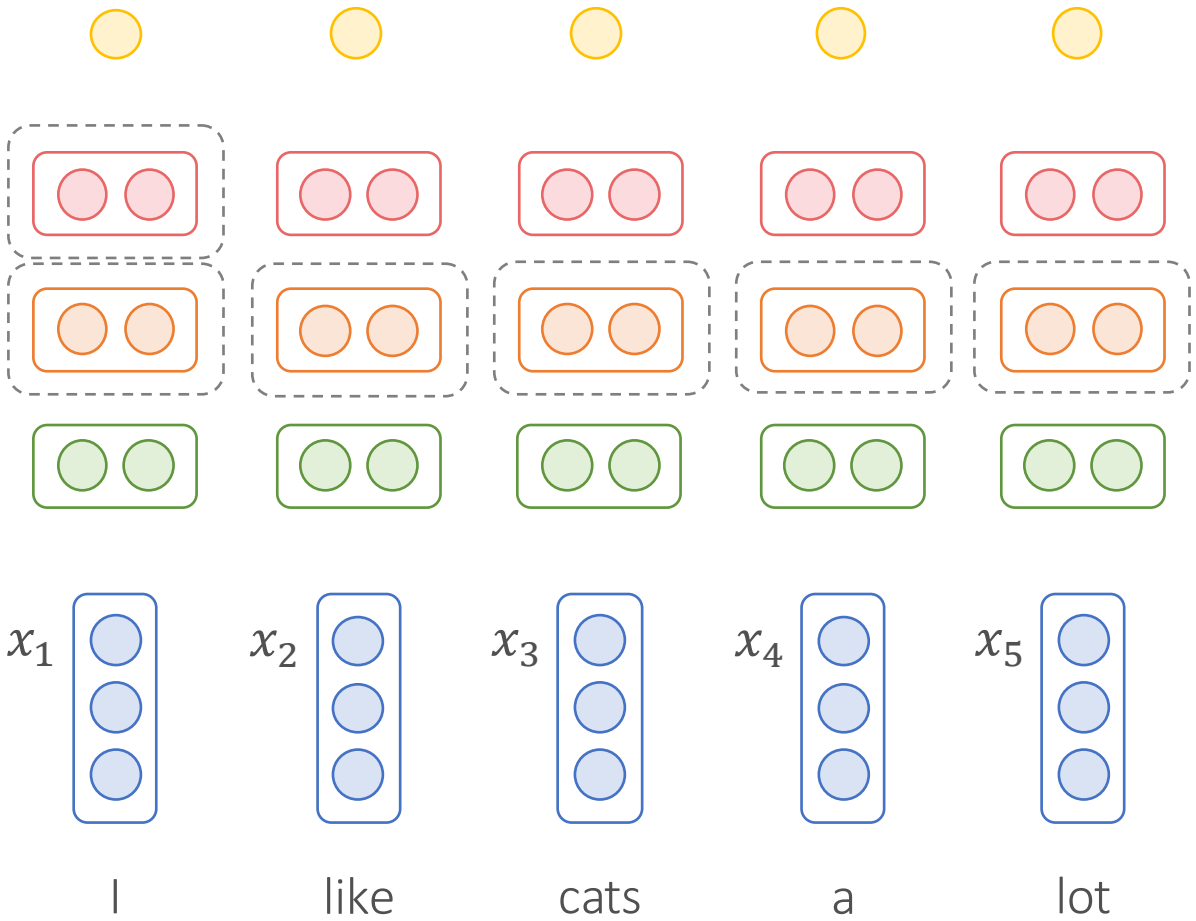
$$\alpha_{m,n} = \text{softmax}\left(\frac{W^Q x_m \cdot W^K\left(x_n + RE(r_{m,n})\right)}{\sqrt{d}}\right)$$

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)

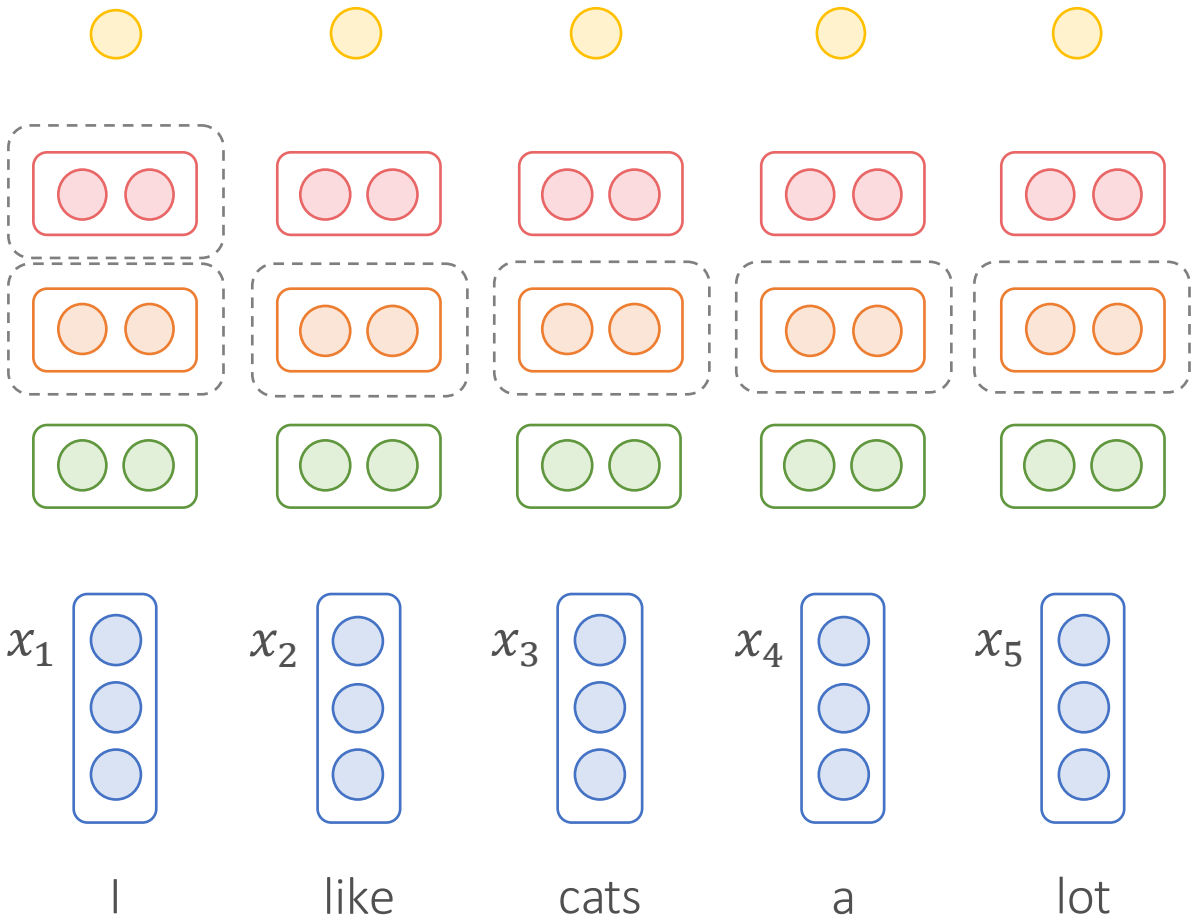
$$\alpha_{m,n} = \text{softmax}\left(\frac{\langle (W^Q x_m) e^{im\theta} \cdot (W^K x_n) e^{in\theta} \rangle}{\sqrt{d}}\right)$$

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 $m\theta$

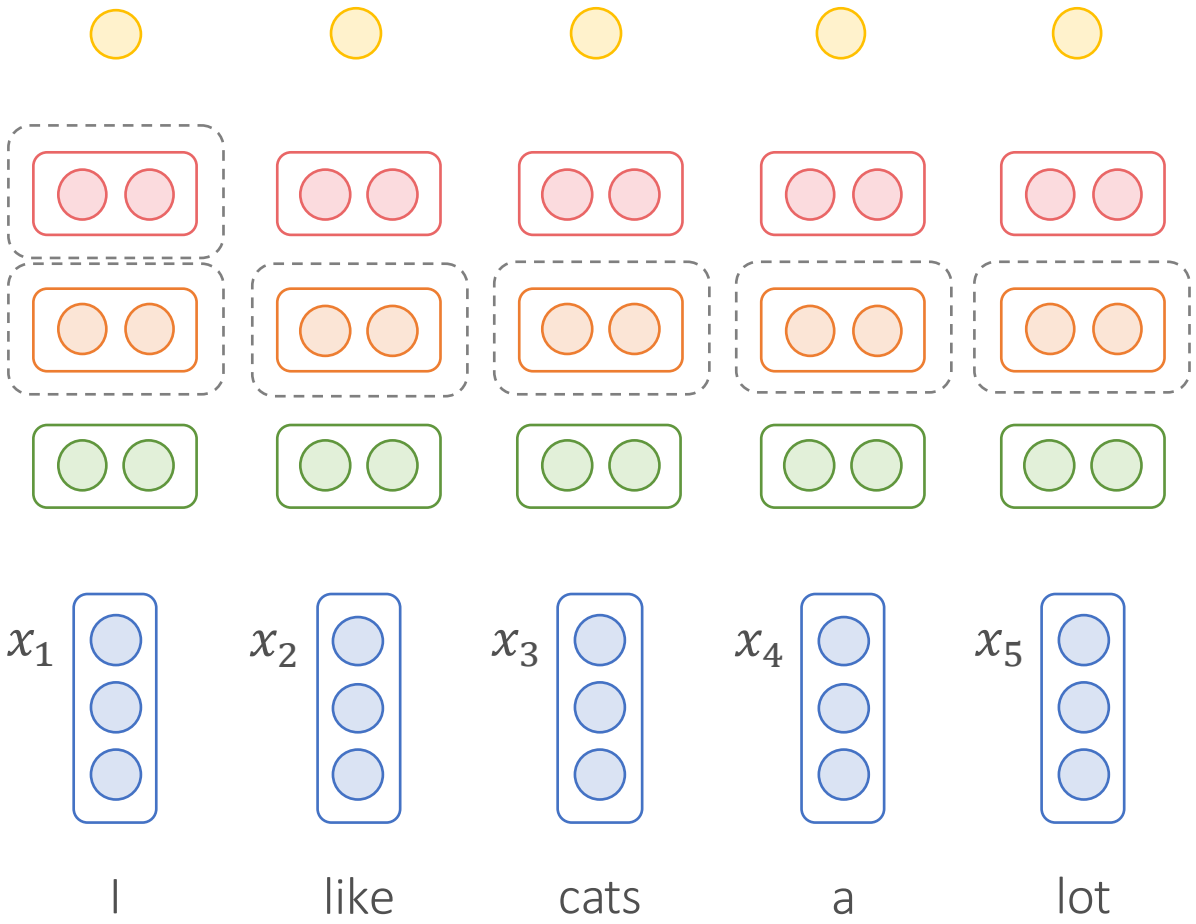
$$\alpha_{m,n} = \text{softmax}\left(\frac{\langle (W^Q x_m) e^{im\theta} \cdot (W^K x_n) e^{in\theta} \rangle}{\sqrt{d}}\right)$$

Normalized Attention Scores

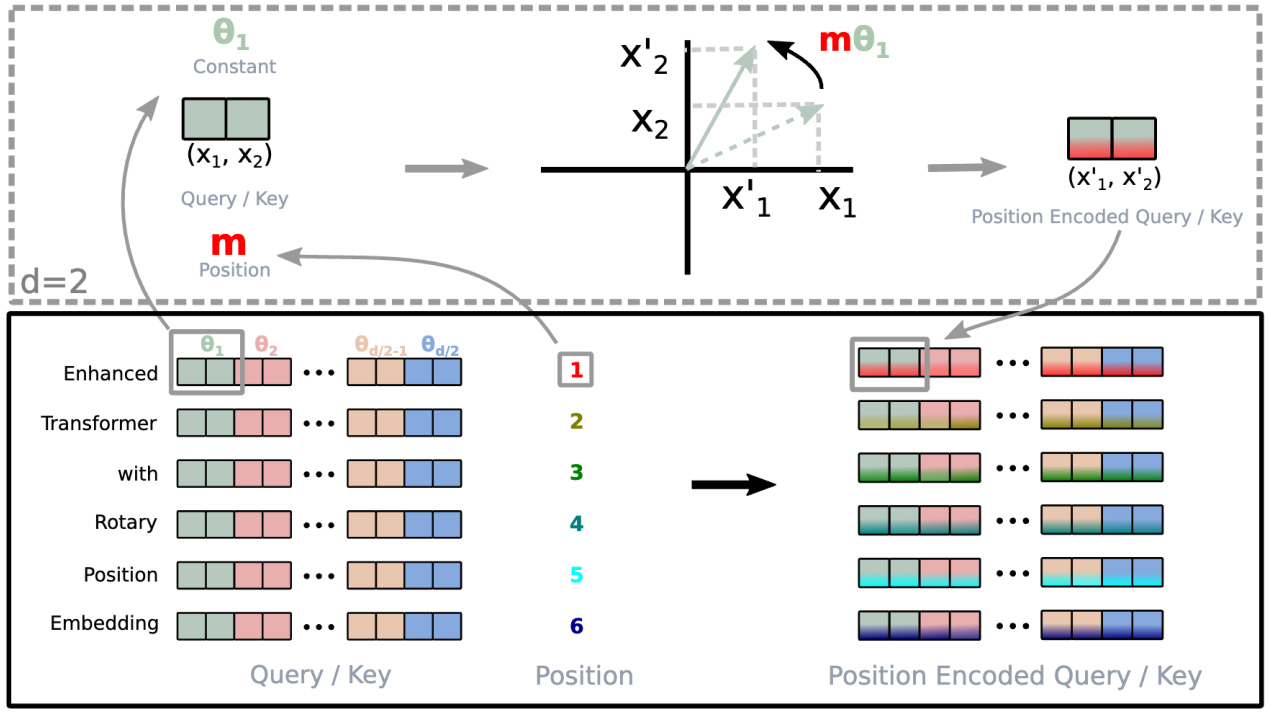
Query $q_i = W^Q x_i$

Key $k_i = W^K x_i$

Value $v_i = W^V x_i$



Self-Attention with RoPE (In 2D Case)



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

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

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

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

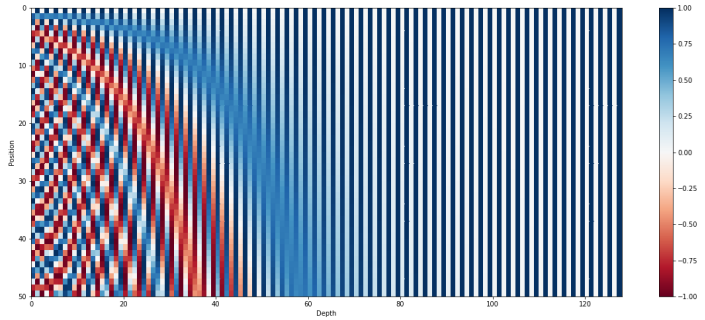
General Form of RoPE

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

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

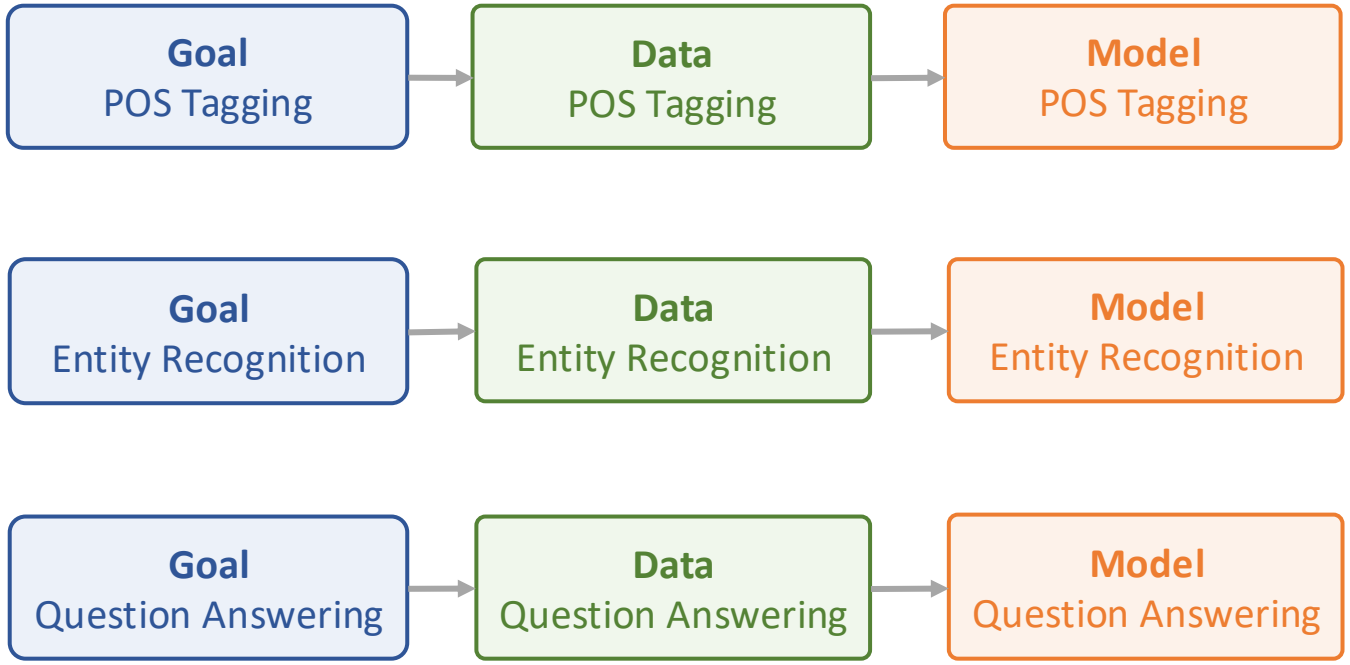
RoPE Performance

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

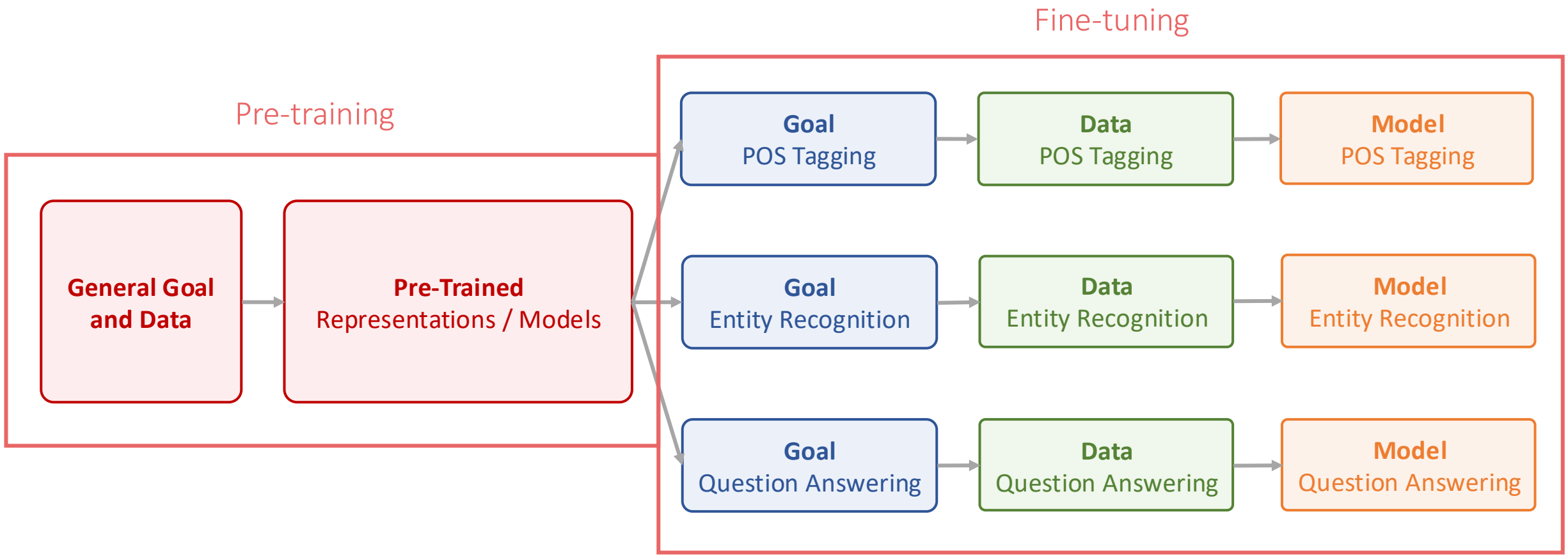
Pre-Training

- Pre-training and fine-tuning
  - First, pre-train a model on a large dataset for task X
  - Them, fine-tune the same on a dataset for task Y
- If task X is somewhat related to task Y
  - Good performance on task X → It is helpful for task Y
- The objective of task X is typically self-supervised
- Word2Vec and ELMo are one kind of pre-training
  - Task X: Predicting context words / Language modeling
  - Task Y: Any downstream tasks

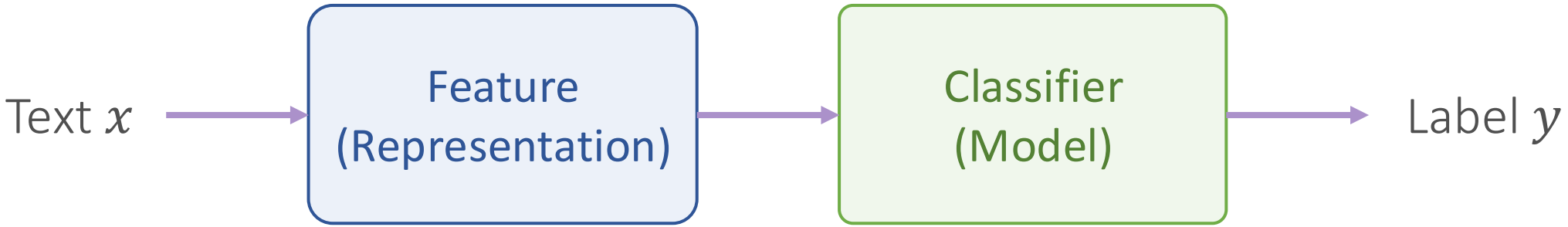
Training from Scratch



Fine-Tuning with Pre-Training

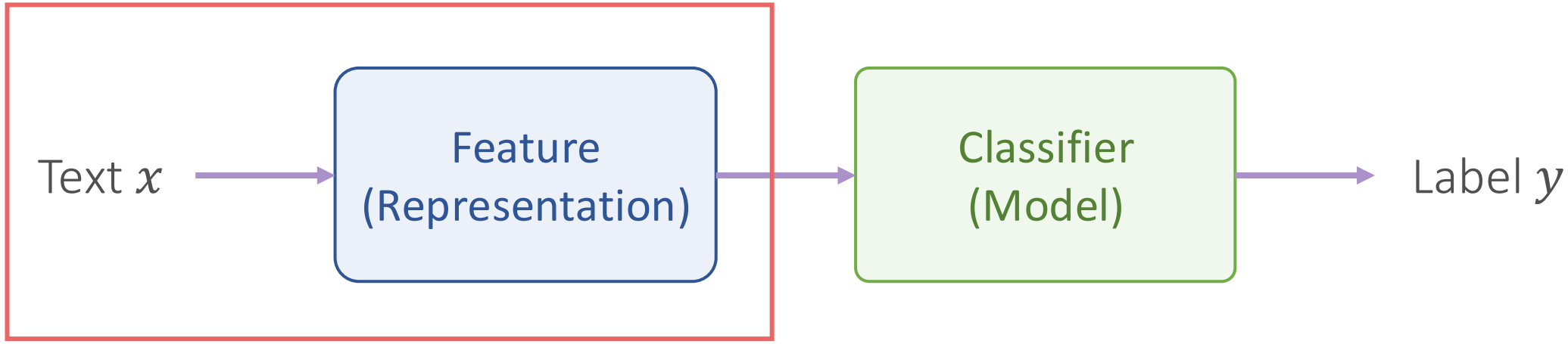


A General Framework for Text Classification



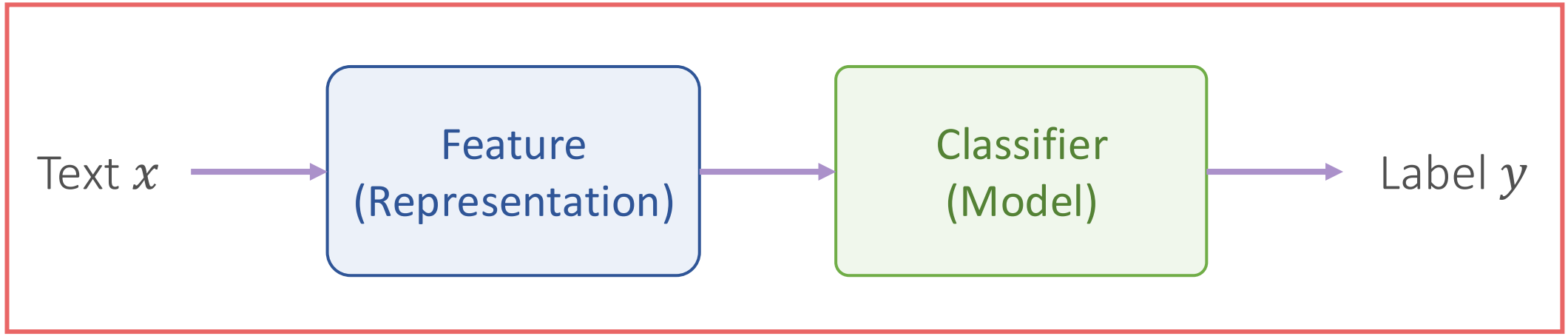
- Task-specific feature: N-gram features, TF-IDF
- Task-specific classifier: Logistic Regression, CNN, RNN, Transformers
- No pre-training

A General Framework for Text Classification



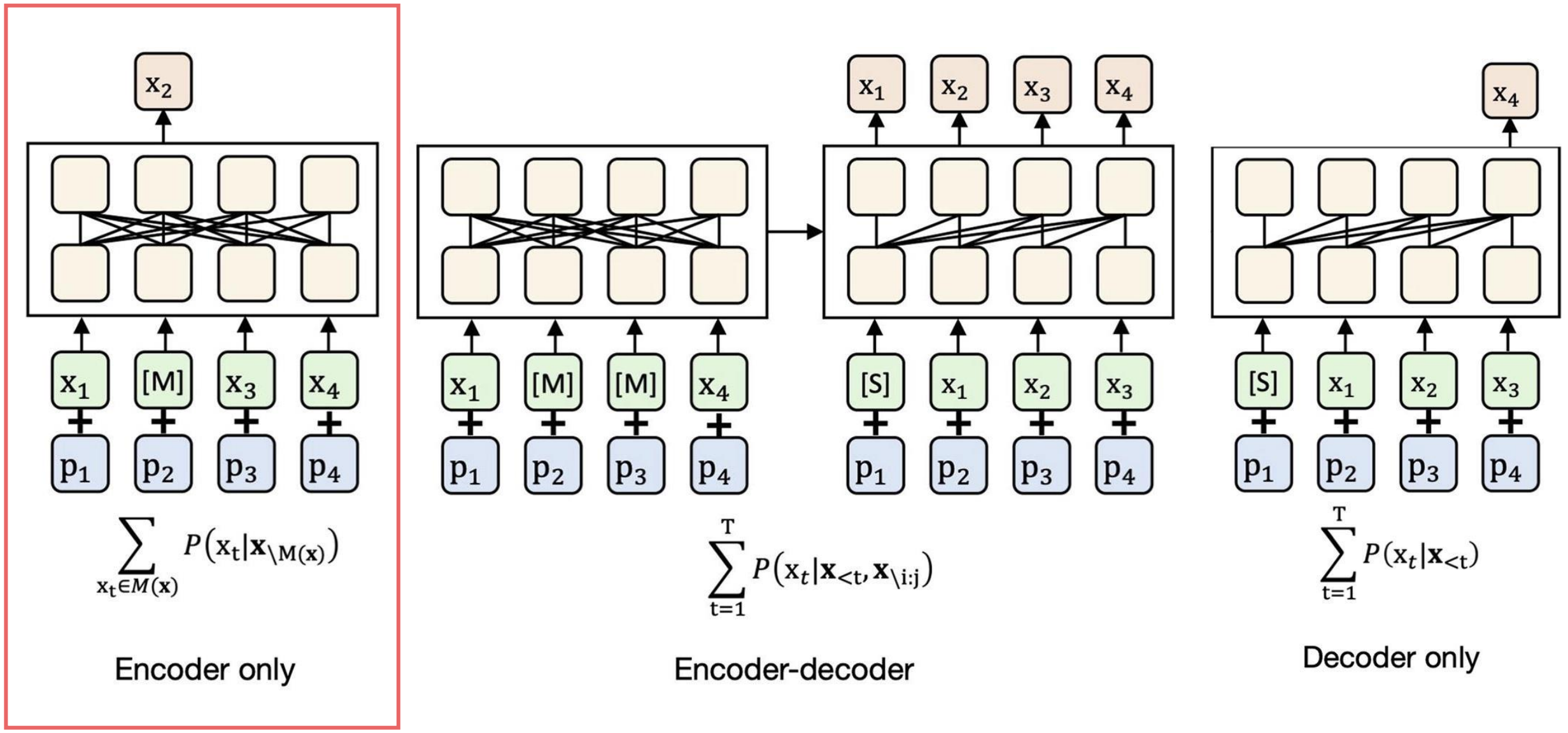
- Pre-trained feature: Word2Vec, Glove
- Task-specific classifier: Logistic Regression, CNN, RNN, Transformers
- Pre-training on features/representations only

A General Framework for Text Classification



- Pre-training the whole pipeline
  - Pre-trained representations + pre-trained model weights
  - We only cover Transformer-based pre-training

Types of Pre-Training



https://www.sciencedirect.com/science/article/pii/S2095809922006324

Encoder-Only: BERT

- **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (BERT)

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

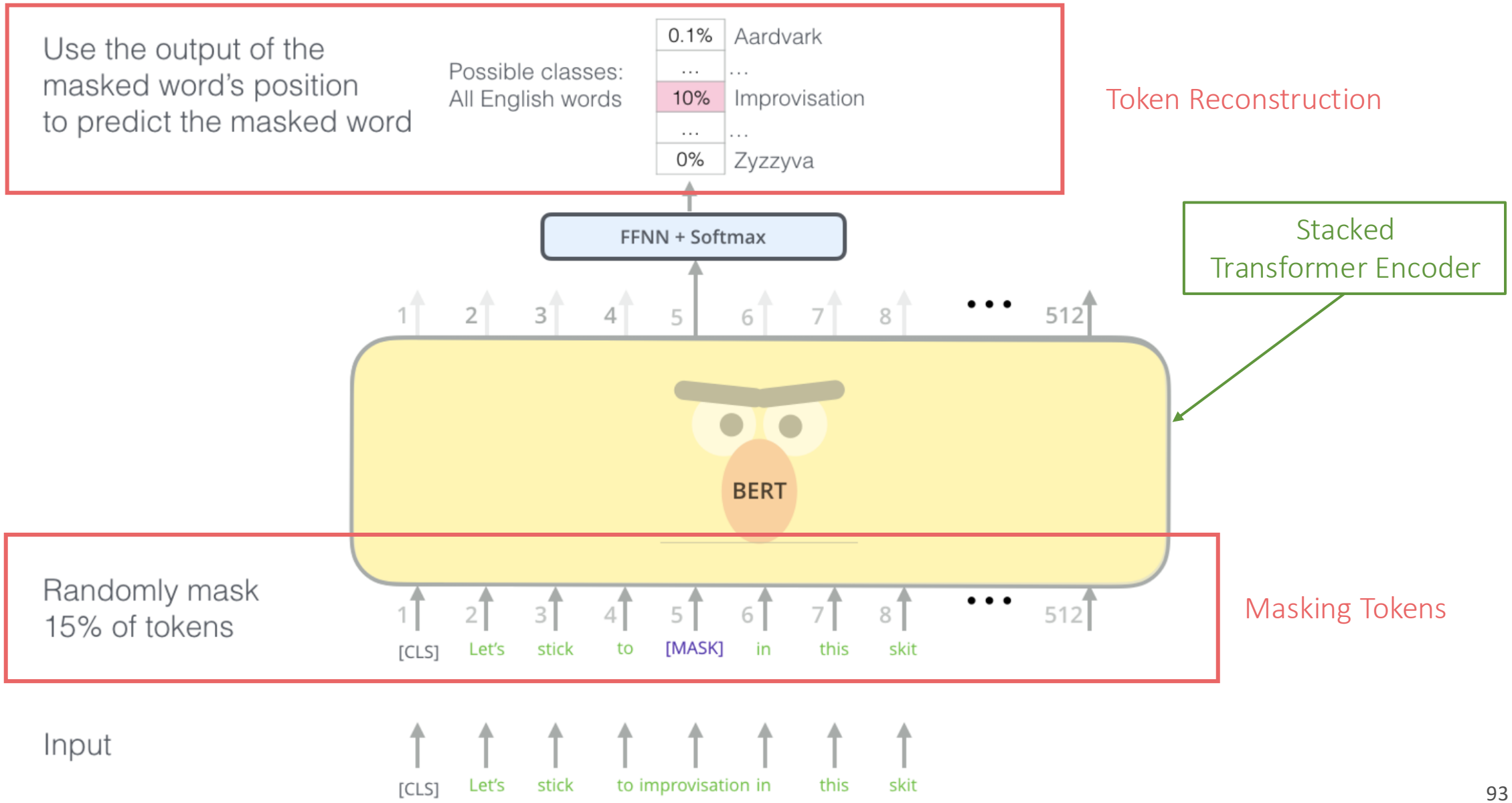
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018

Encoder-Only: BERT

- Transformer architecture
- Encoder-only
  - More about representations
  - Bi-directional
- Pre-training corpus
  - Wikipedia (2.5B tokens) + BookCorpus (0.8B tokens)
- Two self-supervised objectives
  - Masked language modeling
  - Next sentence prediction

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018

Pre-Training Task: Masked Language Modeling



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

Pre-Training Task: Masked Language Modeling

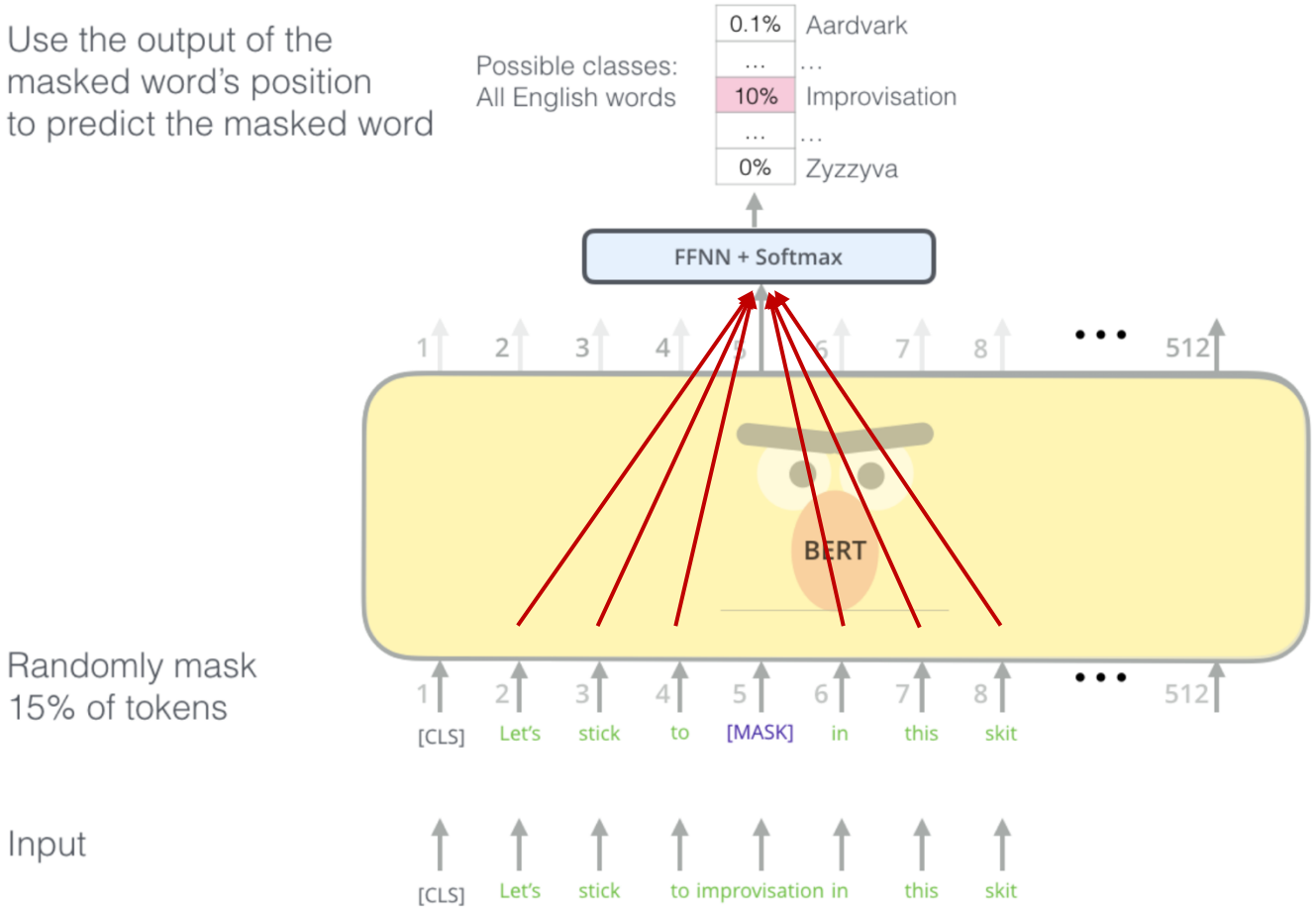
- Why is it useful?
  - Learn to aggregate information from context

Distributional hypothesis: words that occur in similar contexts tend to have similar meanings

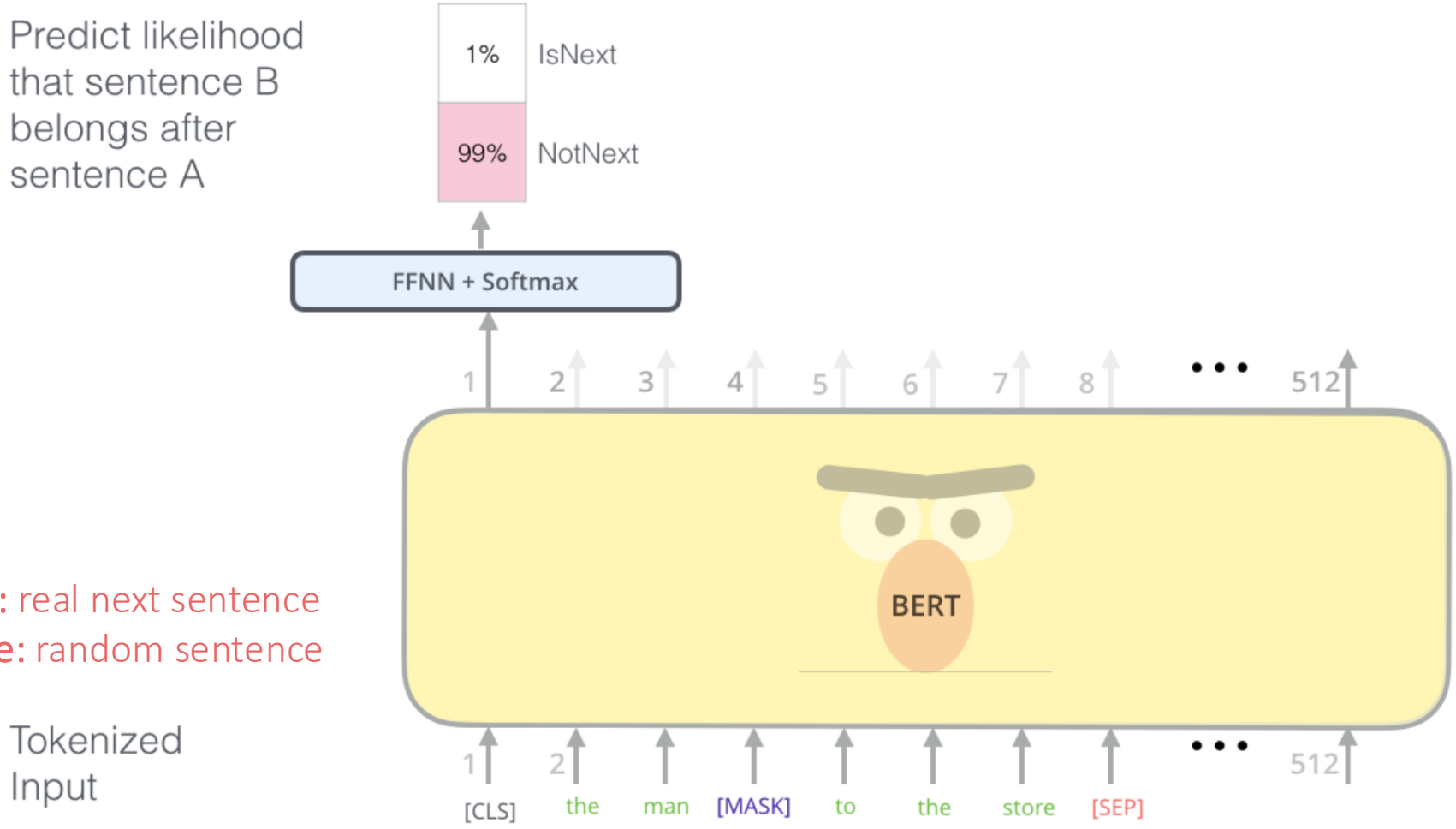
J.R.Firth 1957

- "You shall know a word by the company it keeps"
- One of the most successful ideas of modern statistical NLP!

…government debt problems turning into ***banking*** *crises as happened in 2009…*
…saying that Europe needs unified ***banking*** *regulation to replace the hodgepodge…*
…India has just given its ***banking*** *system a shot in the arm…*



Pre-Training Task: Next Sentence Prediction



Positive example: real next sentence
Negative example: random sentence

Input

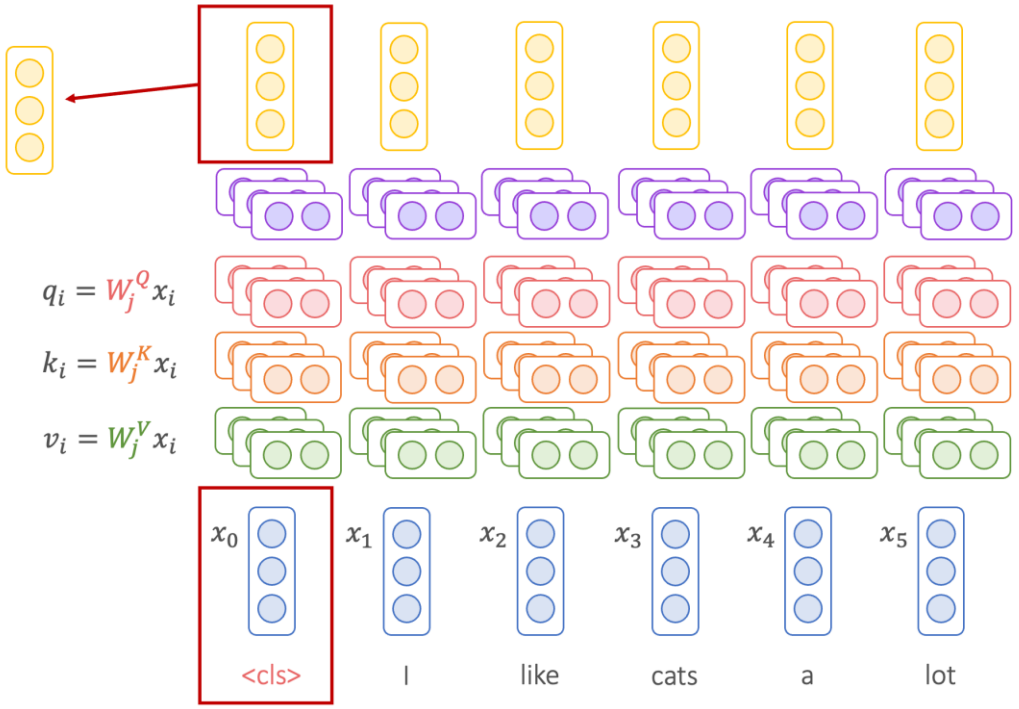
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]

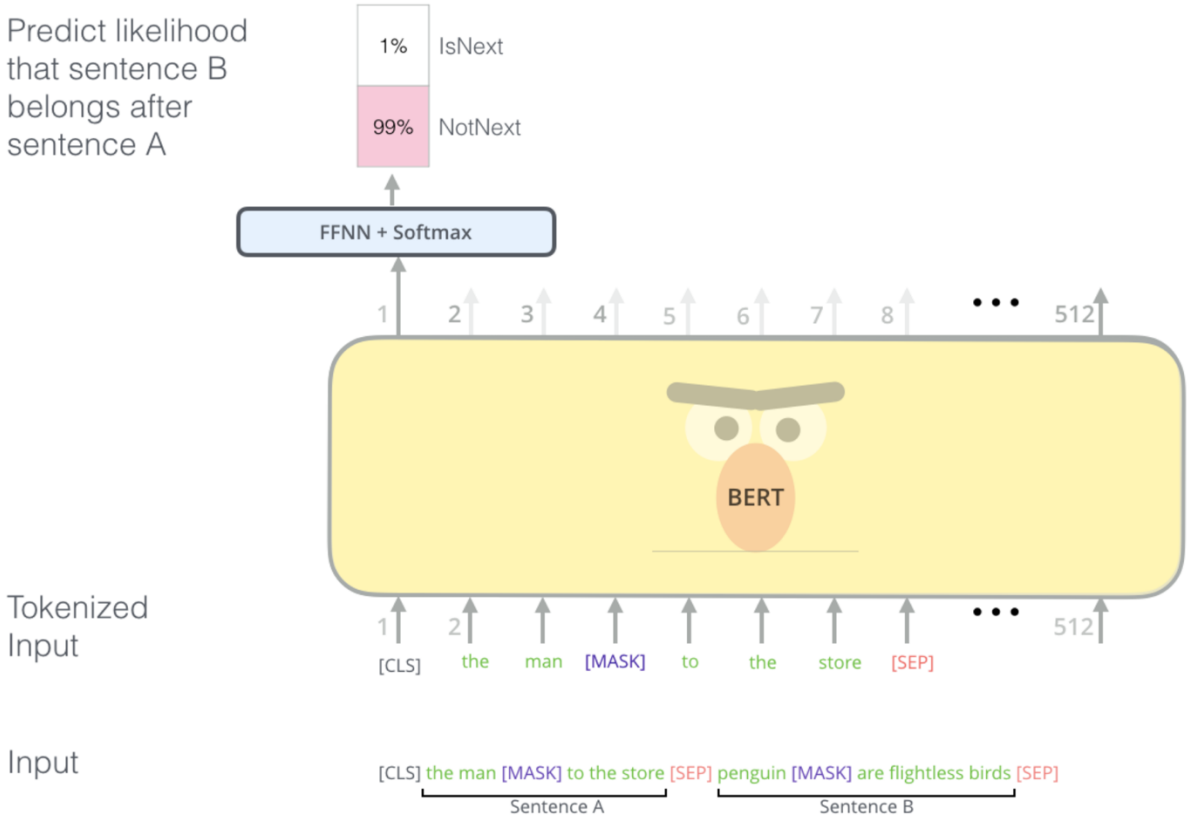
Sentence A | Sentence B

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

Pre-Training Task: Next Sentence Prediction

- Why do we need this?

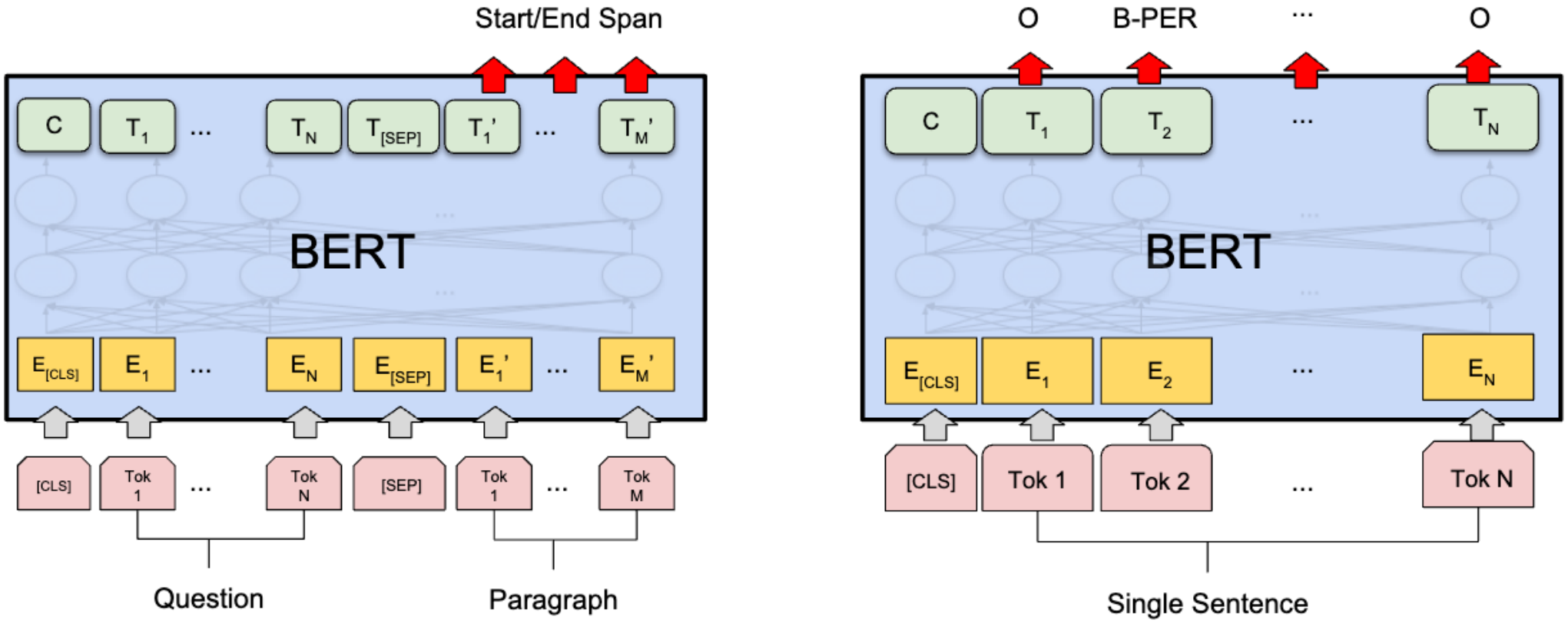




Do we really need this (?)

Fine-Tuning: Token-Level Tasks

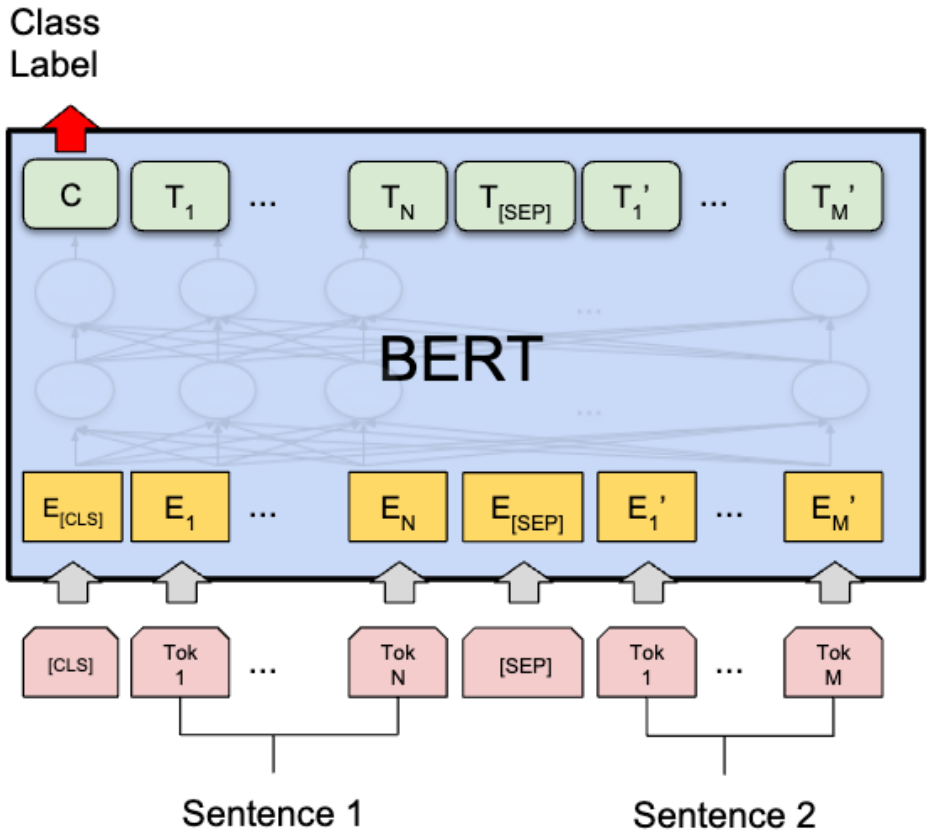
- Pre-training provides a good weight initialization

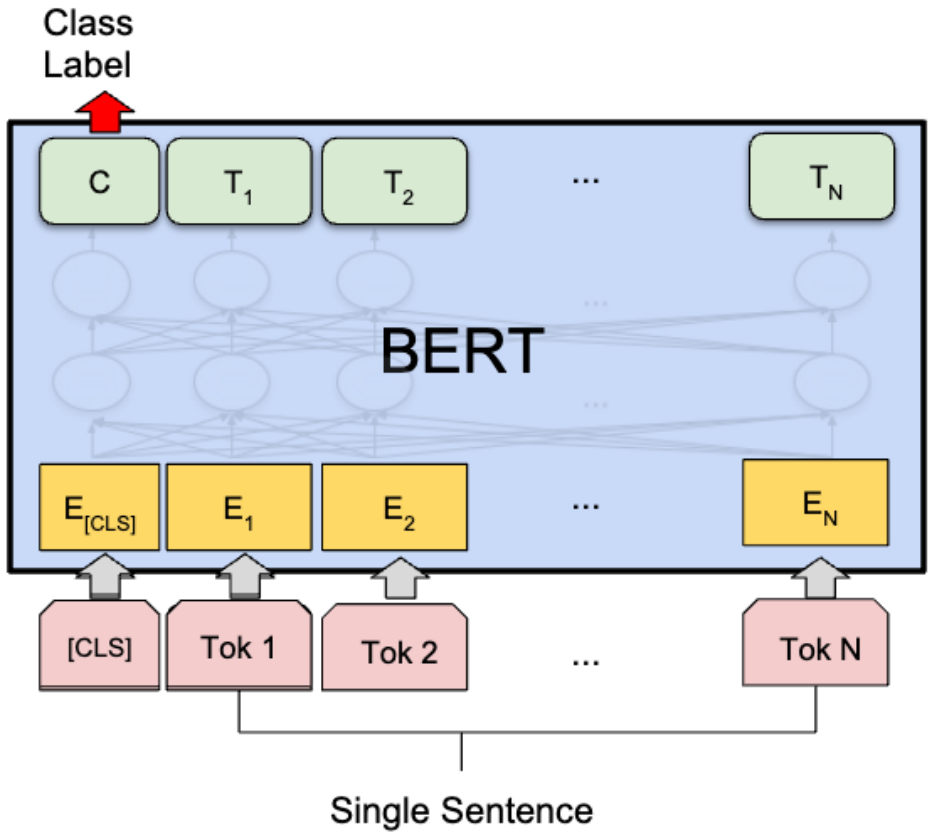


BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018

Fine-Tuning: Sentence-Level Tasks

- Pre-training provides a good weight initialization





BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018

Use BERT

Hugging Face

- BERT-base
  - 12 layers, hidden size = 768, 12 attention heads
  - # parameters ≈ 110M
- BERT-large
  - 24 layers, hidden size = 1024, 16 attention heads
  - # parameters ≈ 340M
- Cased vs. Uncased

Amazing Performance

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

Encoder-Only: RoBERTa

RoBERTa: A Robustly Optimized BERT Pretraining Approach

Yinhan Liu[*§] **Myle Ott**[*§] **Naman Goyal**[*§] **Jingfei Du**[*§] **Mandar Joshi**[†]
Danqi Chen[§] **Omer Levy**[§] **Mike Lewis**[§] **Luke Zettlemoyer**[†§] **Veselin Stoyanov**[§]

[†] Paul G. Allen School of Computer Science & Engineering,
University of Washington, Seattle, WA
{mandar90,lsz}@cs.washington.edu

[§] Facebook AI
{yinhanliu,myleott,naman,jingfeidu,
danqi,omerlevy,mikelewis,lsz,ves}@fb.com

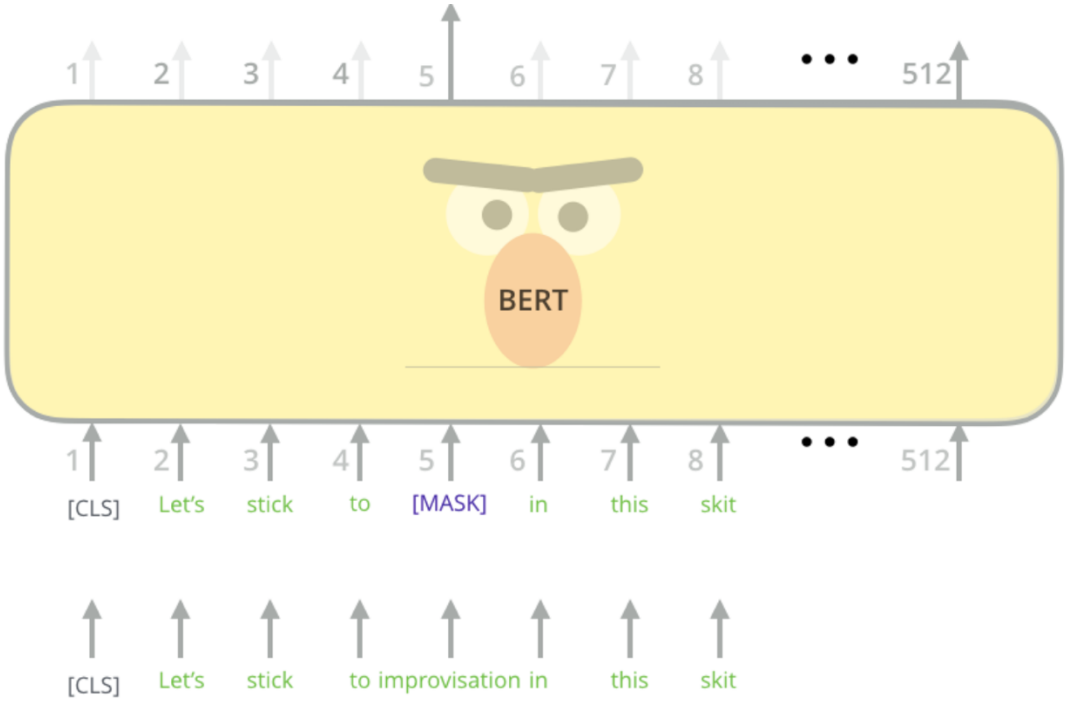
RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019

Encoder-Only: RoBERTa

- **R**obustly **o**ptimized **BERT a**pproach (RoBERTa)
- BERT is still under-trained
- Improve the robustness of training BERT

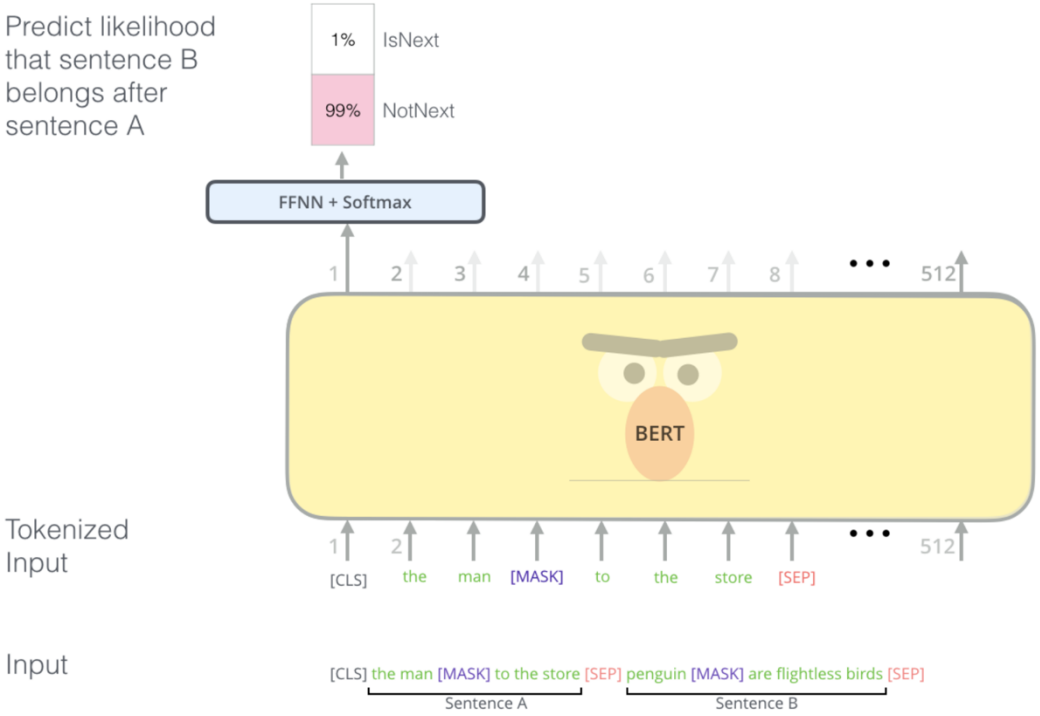
Static Masking vs. Dynamic Masking

- **Static masking:** decide masked words during data pre-processing
- **Dynamic masking**: decide masked words right before feeding into models



Masking	SQuAD 2.0	MNLI-m	SST-2
static	78.3	84.3	92.5
dynamic	78.7	84.0	92.9

Removing Next Sentence Prediction Task



Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation (with NSP loss):				
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation (without NSP loss):				
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6

True Byte-Pair Encoding (BPE)

- BERT: BPE with unicode characters
  - Vocabulary size: 30K
- RoBERTa: BPE with bytes
  - Vocabulary size: 50K

Training Details

- Trained longer
- 10x data
- Bigger batch sizes

Much Better Performance Than BERT

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	**94.6/89.4**	**90.2**	**96.4**
$\text{BERT}_{\text{LARGE}}$						
with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7

Use RoBERTa

Hugging Face

- RoBERTa-base
  - 12 layers, hidden size = 768, 12 attention heads
  - # parameters ≈ 110M
- RoBERTa-large
  - 24 layers, hidden size = 1024, 16 attention heads
  - # parameters ≈ 340M

Questions?