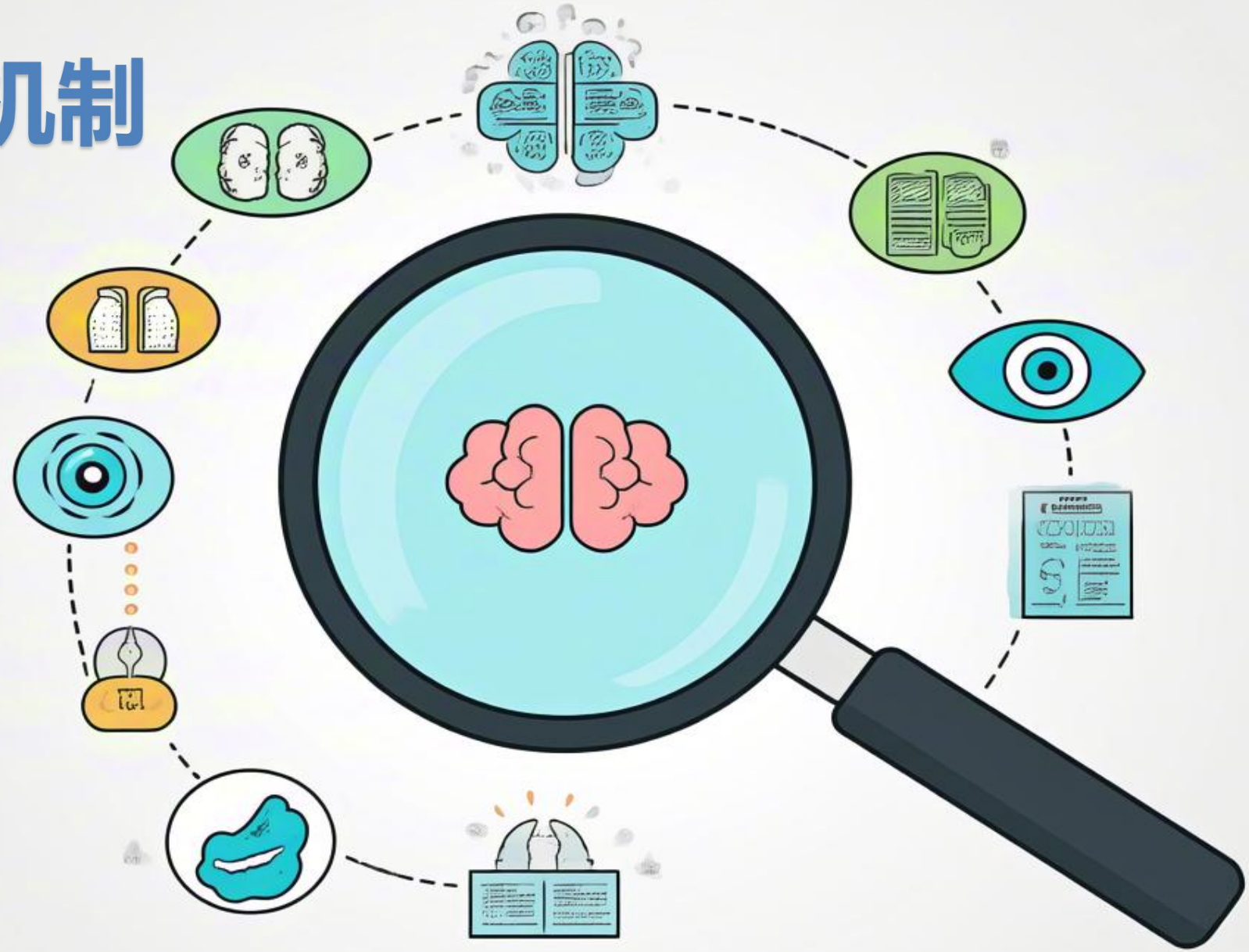
第09讲 注意力机制

欧 新 宇

Thanks to LiMu-CS329 and Hung-yi Lee-MachineLearning

- 注意力机制
- 自注意力
- Transformer

注意力机制



- 什么是注意力机制
- 注意力分数

什么是注意力机制

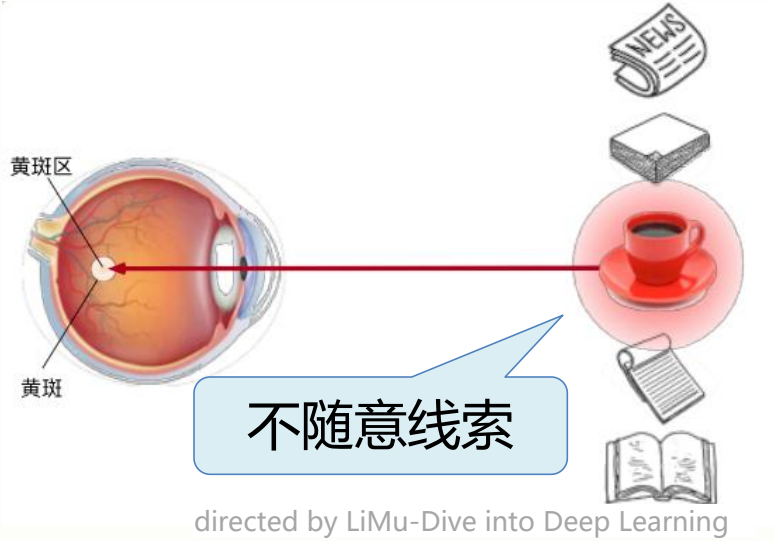
注意力机制

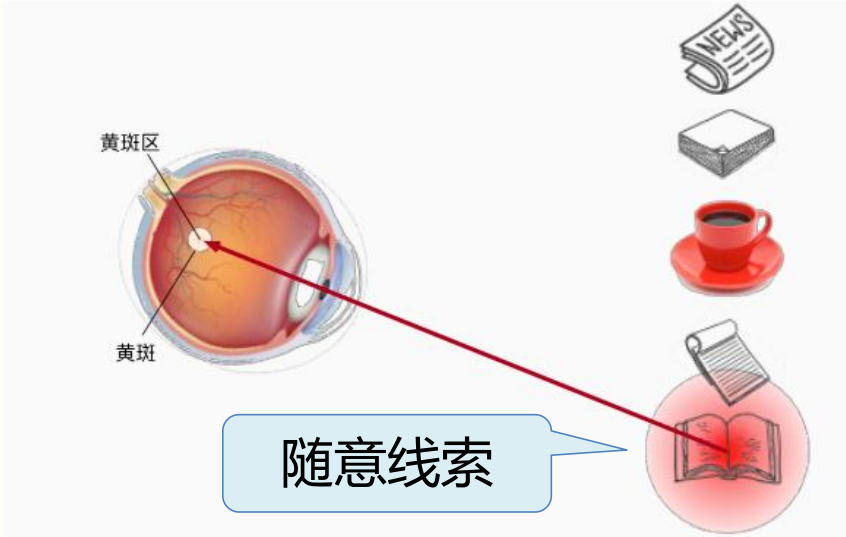
注意力机制的起源

19世纪下半叶，心理学家威廉·詹姆斯在《心理学原理》中提出，注意力是大脑以一种清晰生动的方式在多个同时存在的对象中选择其中一个的过程，它是**意识**对特定对象的**集中与聚焦**。

✓ **注意力** 是人类在长期进化中形成的一种**生存机制**。

✓ **心理学框架：** 生物总是会根据自主的随意线索和非自主的不随意线索选择关注的焦点。

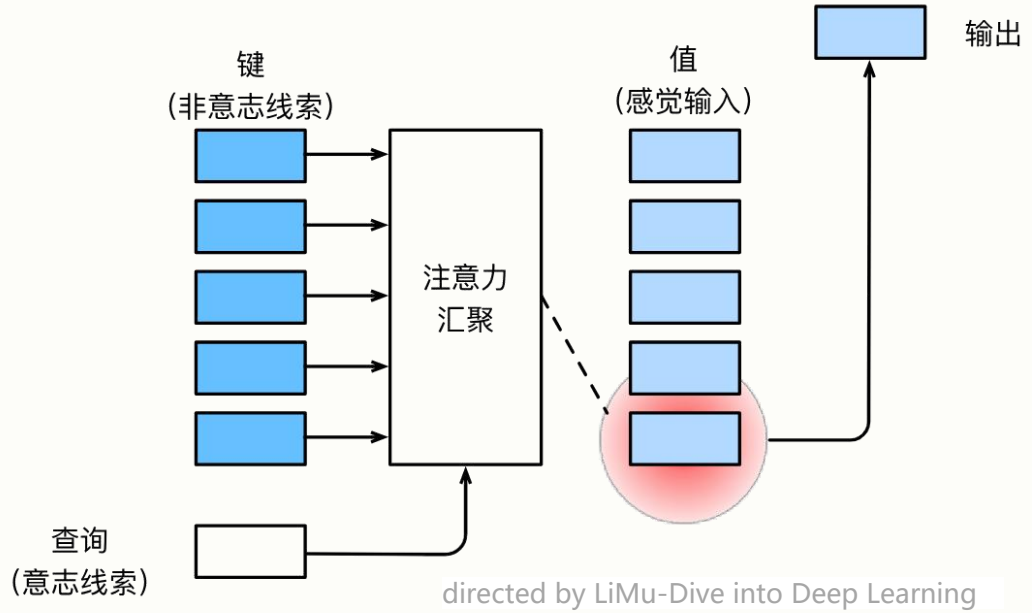




注意力机制

什么是注意力机制

- 卷积、全连接、池化层都只考虑不随意线索
- 注意力机制则显式地考虑随意线索
  - ✓ 随意线索又称为查询（query）
  - ✓ 输入是不随意线索（key）和值（value）组成对
  - ✓ 注意力机制通过注意力汇聚层来有偏向性地选择某些输入



注意力机制

非参注意力池化

- 给定数据对：$(x_i, y_i), i = 1, \ldots, n$
  - ✓ 均值池化：$y = f(x) = \frac{1}{n}\sum_i x_i$
  - ✓ 非参注意力池化（Nadaraya-Watson 核回归）：

query

key

$$f(x) = \sum_{i=1}^{n} \frac{K(x - x_i)}{\sum_{j=1}^{n} K(x - x_j)} y_i$$

value：加权求和

  - ✓ 注意力权重的一般形式：$f(x) = \sum_{i=1}^{n} \alpha(x, x_i)\, y_i$

注意力机制

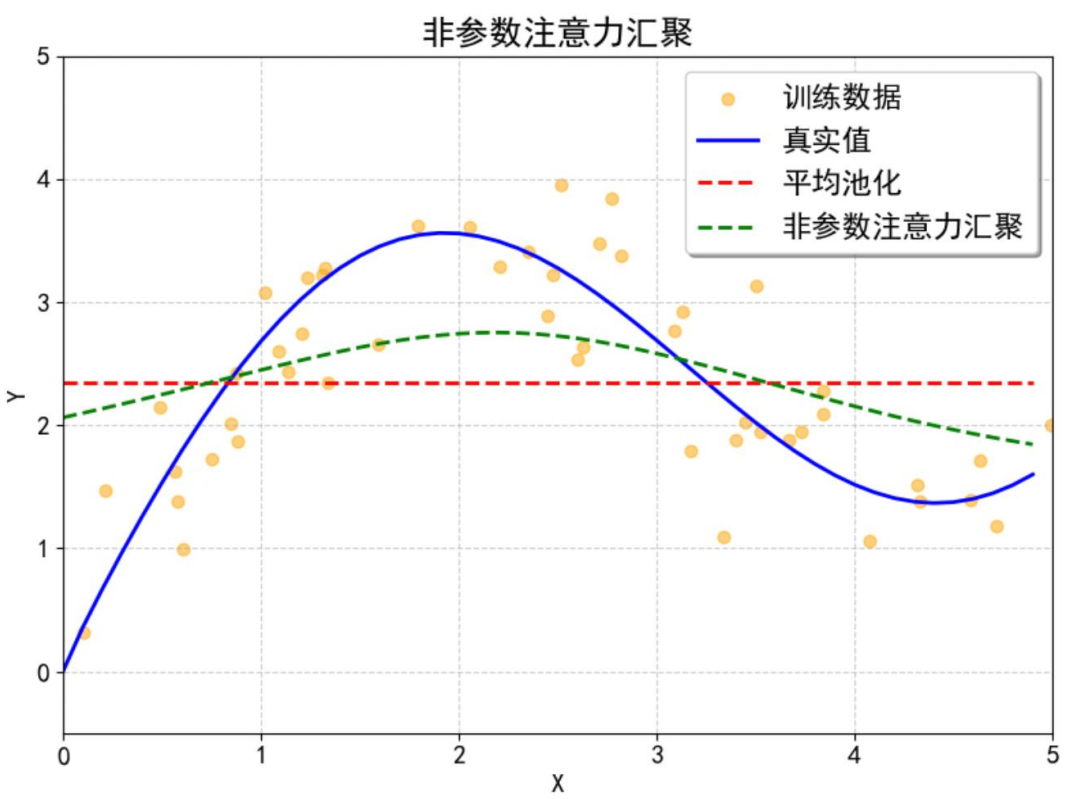
Nadaraya-Watson 核回归

- 高斯核：$K(u) = \frac{1}{\sqrt{2\pi}} exp(-\frac{\mu^2}{2})$
- 基于高斯核的注意力汇聚：

$$f(x) = \sum_{i=1}^{n} \alpha(x, x_i)\, y_i$$

$$= \sum_{i=1}^{n} \frac{\exp(-\frac{1}{2}(x - x_i)^2)}{\sum_{j=1}^{n} \exp(-\frac{1}{2}(x - x_i)^2)} y_i$$

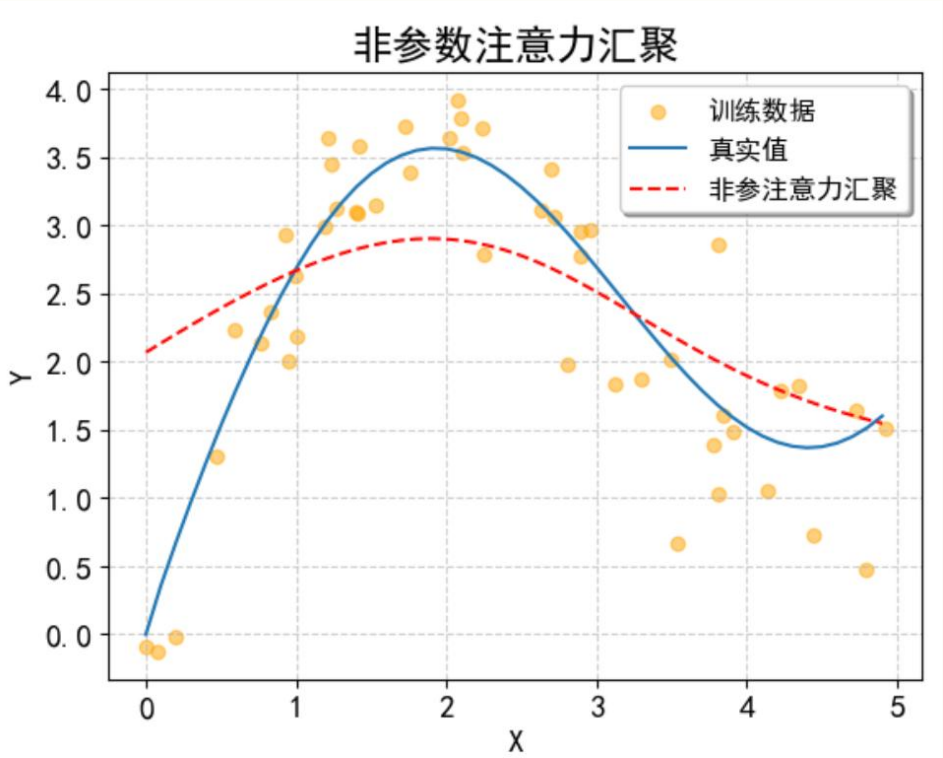
$$= \sum_{j=1}^{n} softmax(-\frac{1}{2}(x - x_i)^2)\, y_i$$

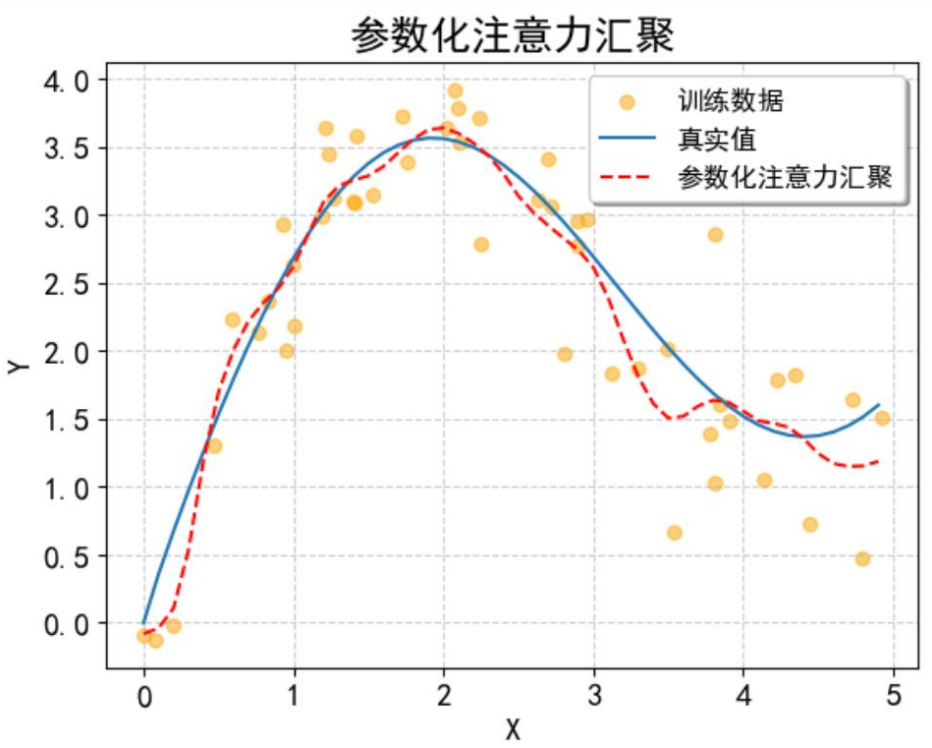


注意力机制

参数化的注意力机制

$$f(x) = \sum_{i=1}^{n} \alpha(x, x_i) w\, y_i = \sum_{j=1}^{n} softmax(-\frac{1}{2}(x - x_i)w^2)\, y_i$$

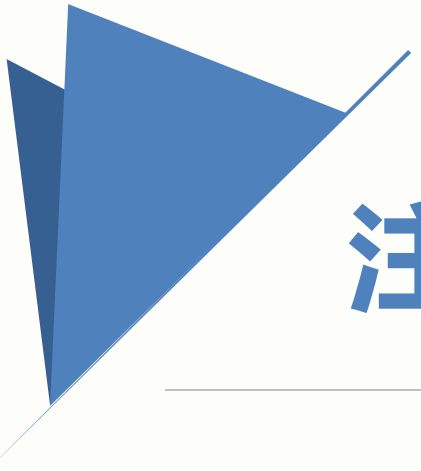




注意力机制

小　结

- 人类的注意力是有限的、有价值和稀缺的资源。
- 心理学：自主性选择的**随意线索**与非自主性感知的**不随意线索**
- 注意力机制与全连接层或者汇聚层的主要区别在于**增加**了**自主提示**，它通过注意力使选择偏向于更有意义的值（value，感官输入），它来源于查询 （query，自主性提示）和键（key，非自主性提示）的交互。
- 注意力机制的一般形式 $f(x) = \sum_{i=1}^{n} a(x, x_i)\, y_i$，其中，$a(x, x_i)$ 是注意力权重
- 注意力机制实现了“既凭直觉抓取模式，又按目标筛选信息”的平衡



注意力分数

注意力分数

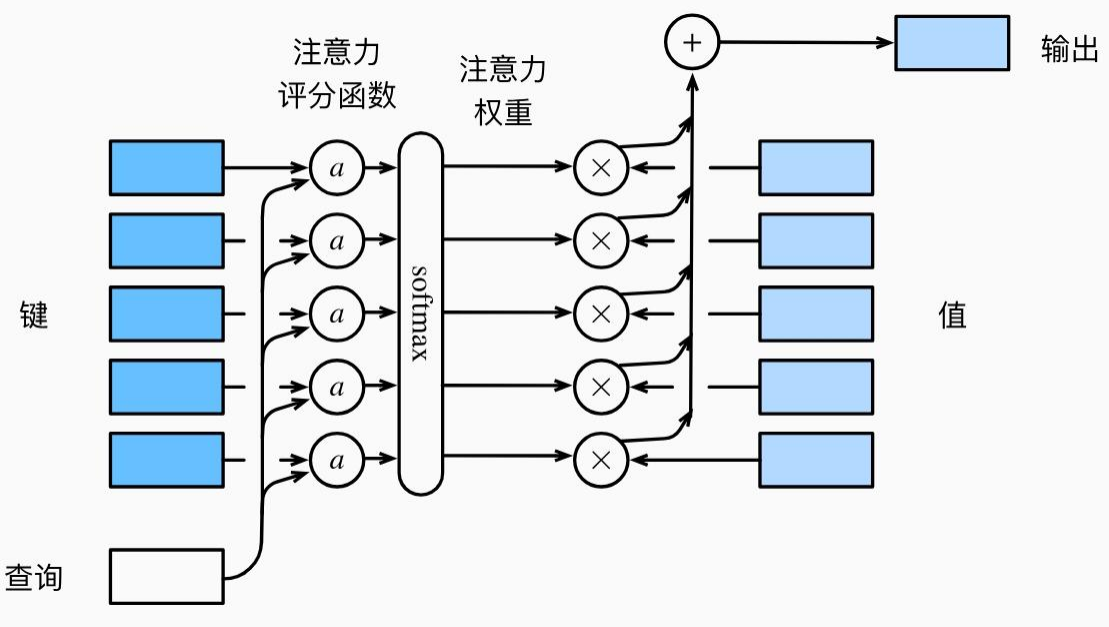
回顾：

注意力分数

注意力权重

注意力分数

$$f(x)=\sum_{i=1}^{n}\alpha(x,x_i)\,y_i=\sum_{j=1}^{n}\mathrm{softmax}(-\frac{1}{2}(x-x_i)^2)\,y_i$$



注意力分数

高维度注意力

假设 query $\boldsymbol{q} \in \mathbb{R}^q$，$m$ 对 key-value $(\boldsymbol{k_1}, \boldsymbol{v_1}),\ldots.$，这里 $\boldsymbol{k}_i \in \mathbb{R}^k$，$\boldsymbol{v}_i \in \mathbb{R}^v$

则注意力池化层变为：

$$f(\boldsymbol{q}, (\boldsymbol{k_1}, \boldsymbol{v_1}), \ldots, \boldsymbol{k_m}, \boldsymbol{v_m}) = \sum_{i=1}^{n} \boldsymbol{\alpha}(\boldsymbol{q}, \boldsymbol{k_i})\, \boldsymbol{v}_i \in \mathbb{R}^v$$

$$\boldsymbol{\alpha}(\boldsymbol{q}, \boldsymbol{k_i}) = softmax(\boldsymbol{\alpha}(\boldsymbol{q}, \boldsymbol{k_i})) = \frac{\exp(\boldsymbol{\alpha}(\boldsymbol{q}, \boldsymbol{k_i}))}{\sum_{j=1}^{n} \exp(\boldsymbol{\alpha}(\boldsymbol{q}, \boldsymbol{k_i}))} \in \mathbb{R}$$

注意力分数

注意力分数

加性注意力（Additive Attention）

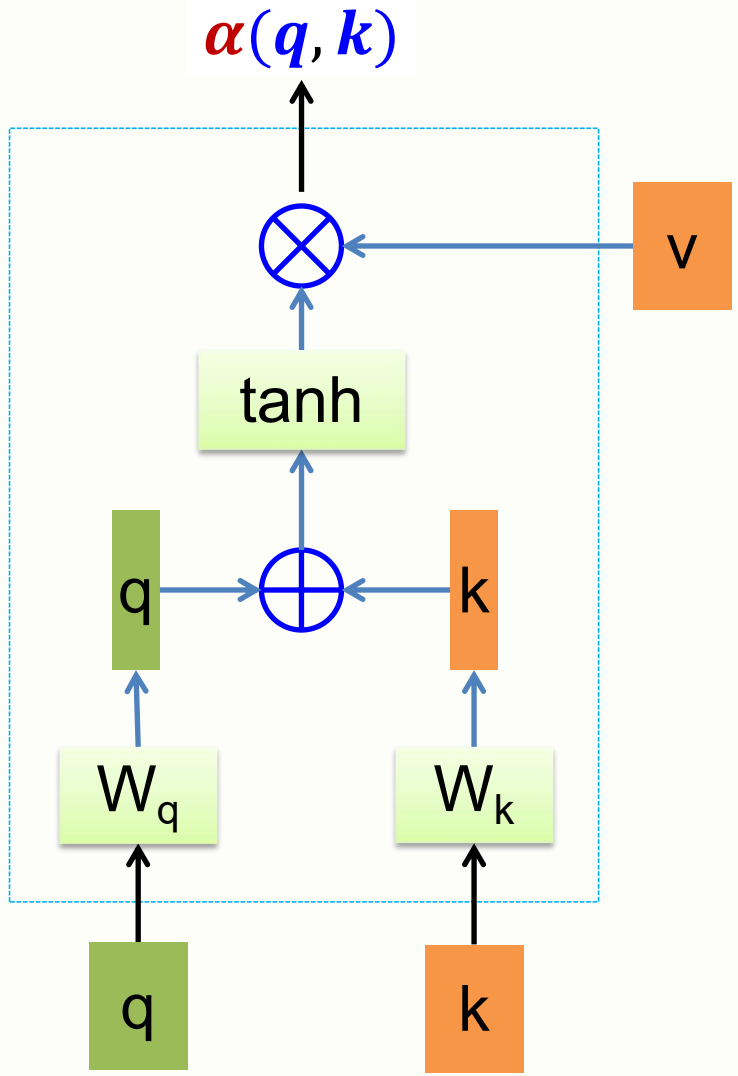
- 给定查询向量 q，键向量 k
- 可学习参数：$\boldsymbol{W}_k \in \mathbb{R}^{h\times k}, \boldsymbol{W}_q \in \mathbb{R}^{h\times q}, \boldsymbol{v} \in \mathbb{R}^h$
- 相似度得分：

加权求和　　加性融合

$$\boldsymbol{\alpha}(\boldsymbol{q},\boldsymbol{k}) = \boldsymbol{v}^T \tanh(\boldsymbol{W}_k\boldsymbol{k} + \boldsymbol{W}_q\boldsymbol{q})$$

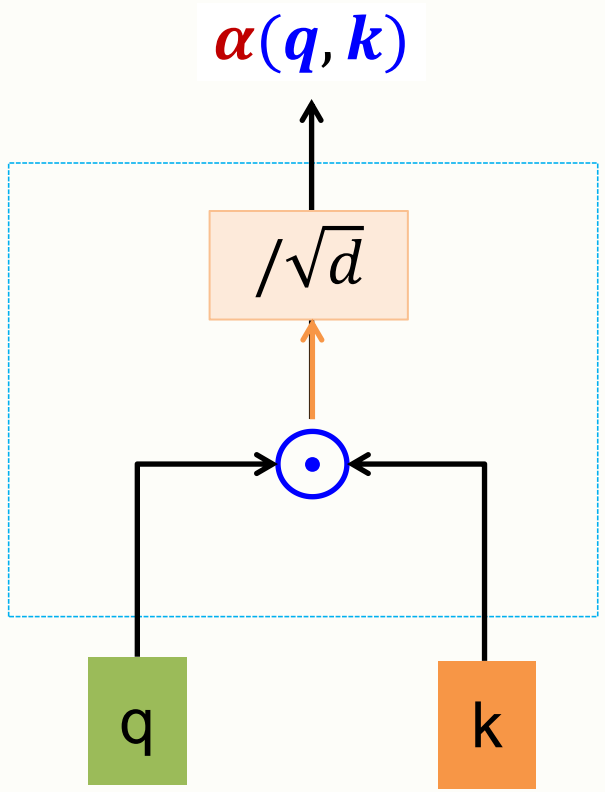
非线性激活

- 等价于一个大小为 h，输出为 1 的单隐层 MLP
- 支持不同维度且需要复杂交互的场景
- 表达能力**强**，但计算开销**大**



注意力分数

缩放点积注意力（Scaled Dot-Product Attention）



当 query 和 key 的长度相等时，且$\boldsymbol{q}, \boldsymbol{k_i} \in \mathbb{R}^d$，那么注意力分数可以表示为：

$$\alpha(\boldsymbol{q}, \boldsymbol{k_i}) = < \boldsymbol{q}, \boldsymbol{k_i} > /\sqrt{d}$$

向量化版本：

- ✓ $\mathbf{Q} \in \mathbb{R}^{n\times d}, \boldsymbol{K}_q \in \mathbb{R}^{m\times d}, \boldsymbol{V} \in \mathbb{R}^{m\times d}$
- ✓ 注意力分数：$\boldsymbol{\alpha}(\boldsymbol{Q}, K) = \boldsymbol{Q}\boldsymbol{K}^T/\sqrt{d} \in \mathbb{R}^{n\times m}$
- ✓ 注意力权重：

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}}/)\boldsymbol{V} \in \mathbb{R}^{n\times v}$$

加权求和

缩放因子

注意力分数

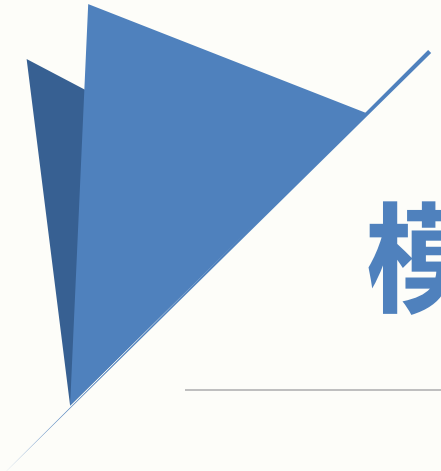
小结

- 注意力分数衡量了查询向量（query）与键向量（key）之间的**相似度**，它反映的是模型对不同键值对的**关注程度**。
- Softmax 实现将注意力分数转化为更加**稳定**的**注意力权重**，确保了模型的有效训练。
- 加性注意力（Additive Attention）和缩放点积注意力（Scaled Dot-Product Attention）是两种常见的注意力分数计算方法。
  - ✓ **加性注意力：**将 query 和 key 合并起来进入一个单输出单隐藏层的MLP，适用于任务复杂、且需要强拟合的场景
  - ✓ **缩放点积注意力：**直接将 query 和 key 做内积，复杂度低，但更高效。

自注意力机制

Self-Attention Mechanism

- 模型的输入和输出
- 自注意力
- 位置编码

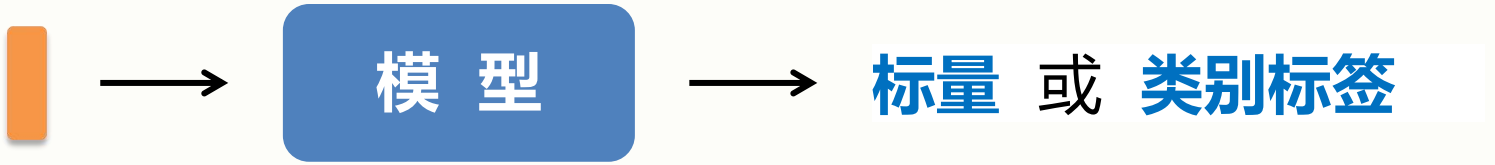


模型的输入和输出

自注意力机制

模型的输入

- ***输入是一个向量***



- ***输入是一组向量***

自注意力机制

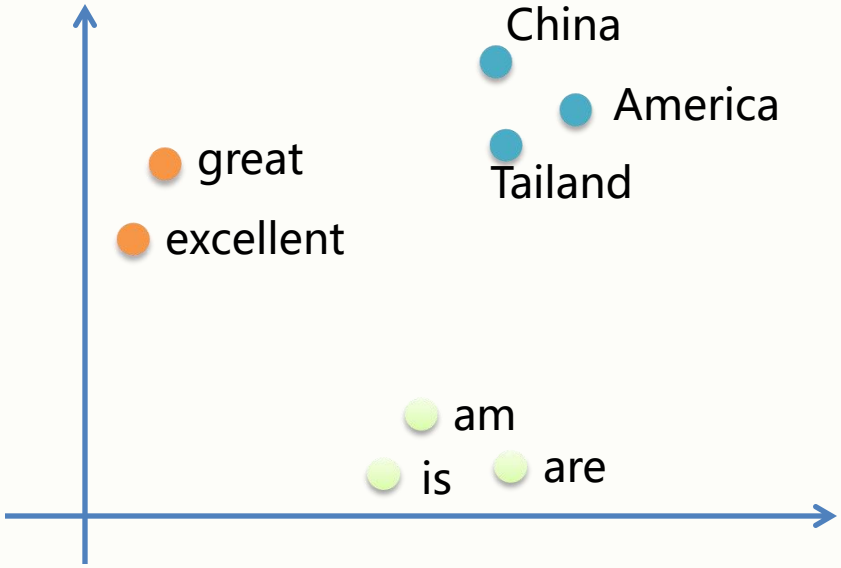
基于向量集（Vector Set）的文本

China is a great country

One - hot 编码

China = [1 0 0 0 0 0 0 ...]
is = [0 1 0 0 0 0 0 ...]
a = [0 0 1 0 0 0 0 ...]
great = [0 0 0 1 0 0 0 ...]
country = [0 0 0 0 1 0 0 ...]
America = [0 0 0 0 0 1 0 ...]
Tailand = [0 0 0 0 0 0 1 ...]

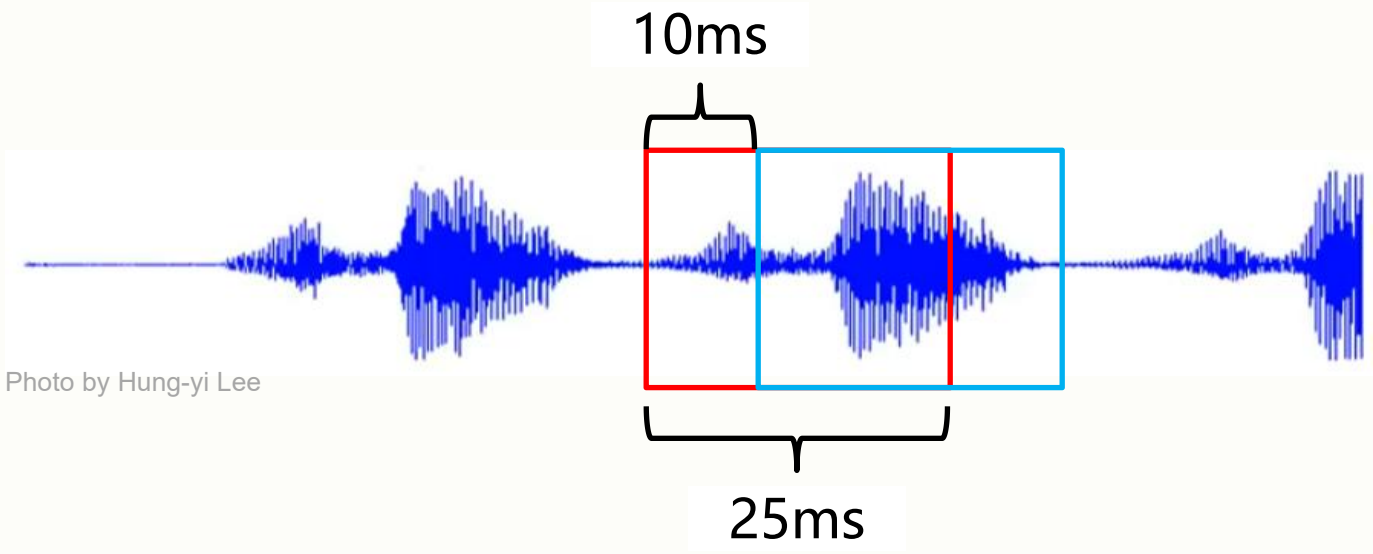
词嵌入（word embedding）



[arXiv1301.3781] Efficient Estimation of Word Representations in Vector Space

自注意力机制

基于向量集的语音



1sec → 100帧

1min → 6000帧

帧向量

400 个采样点（16KHz）

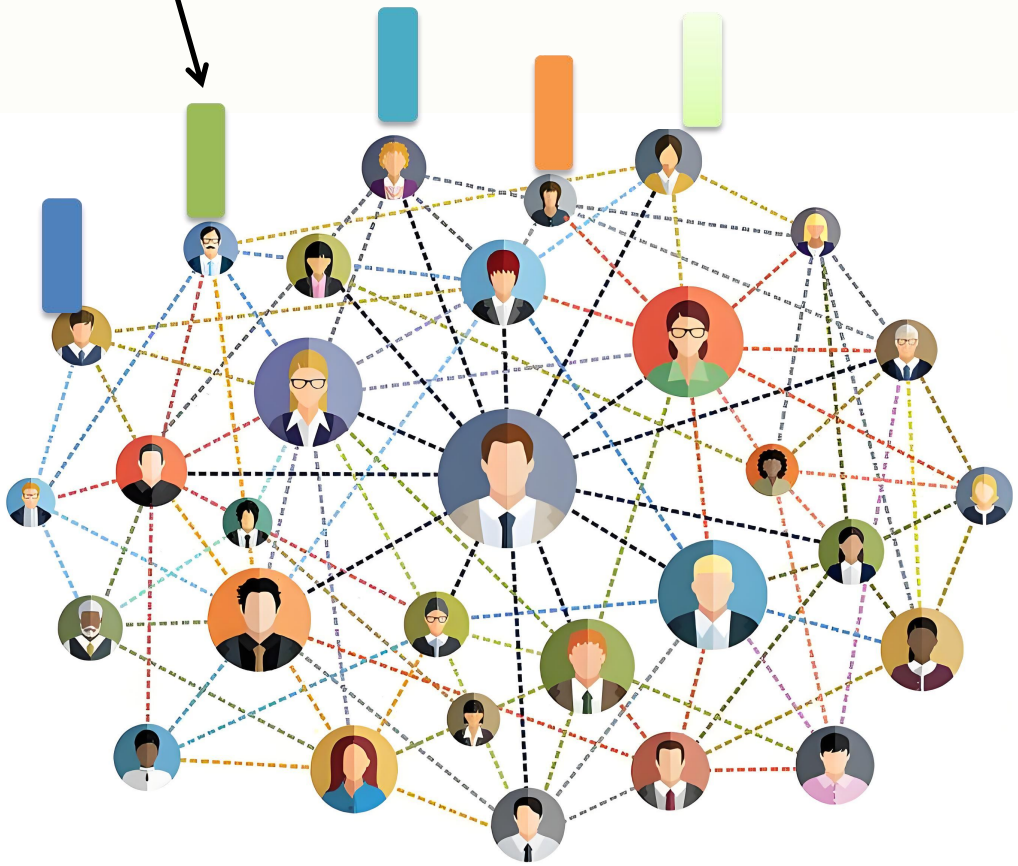
39 维 MFCC

80 维 过滤器组输出

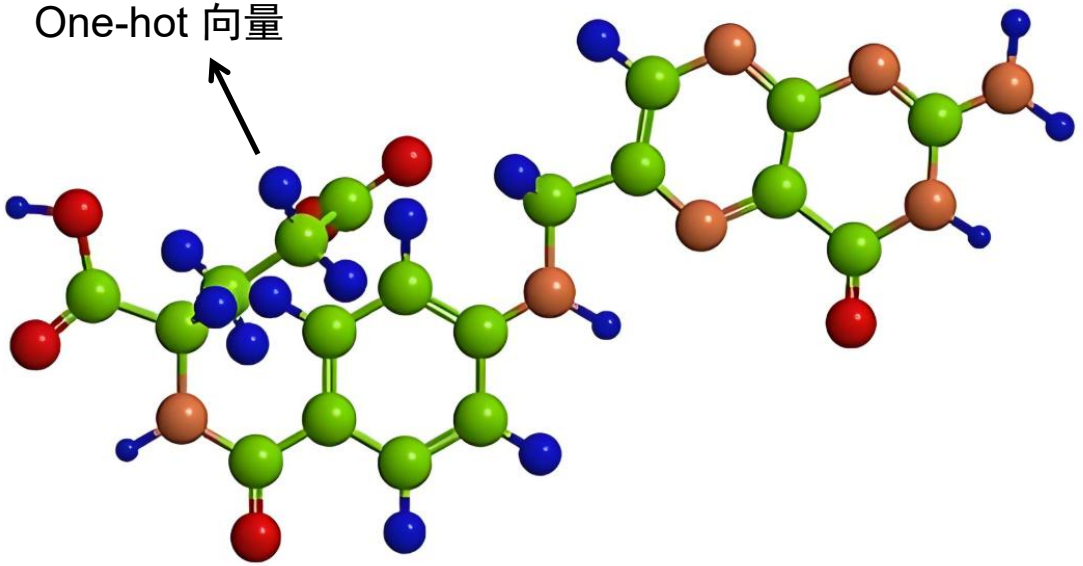
自注意力机制

基于向量集的图网络

每个人用一个向量表示



One-hot 向量



每个原子用一个向量表示

H = [1 0 0 0]

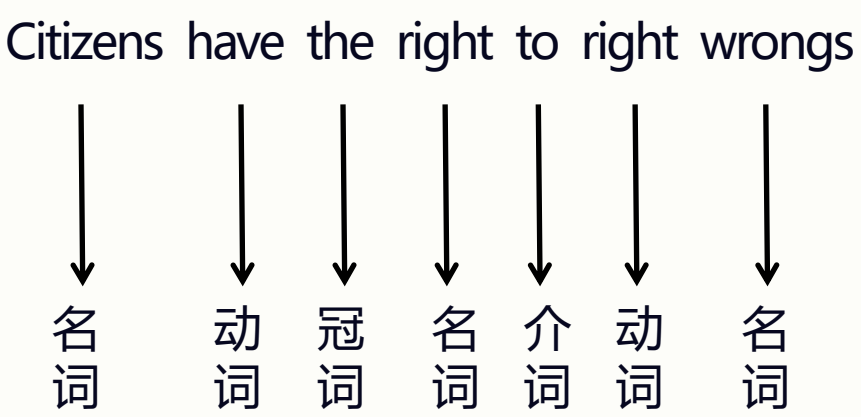
C = [0 1 0 0]

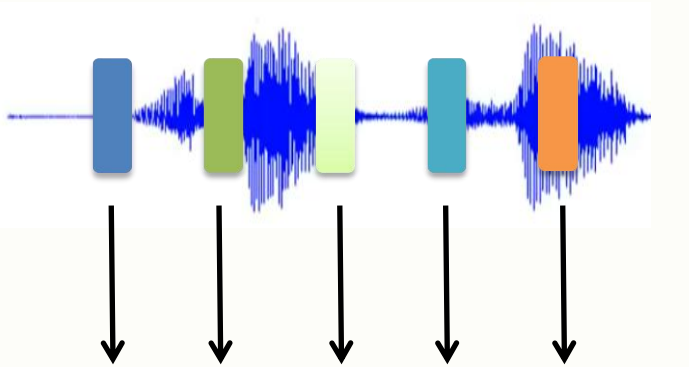
O = [0 0 1 0]

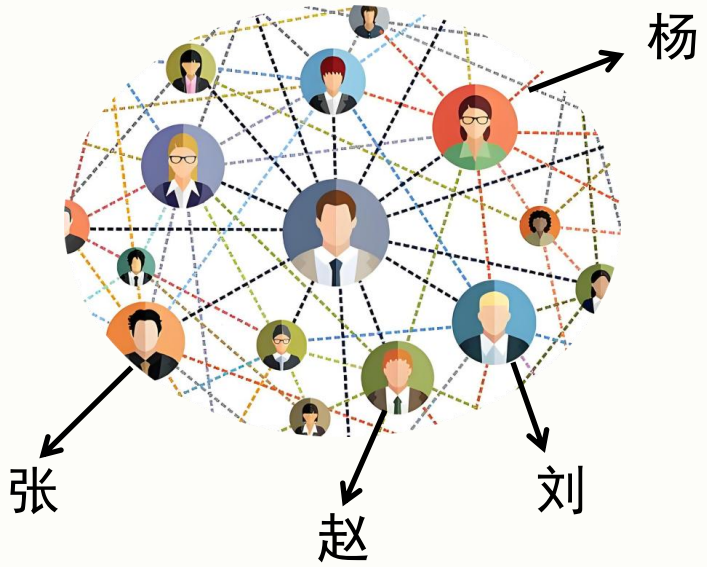
自注意力机制

多种多样的输入

- *每个向量输出一个结果（等长输出）*





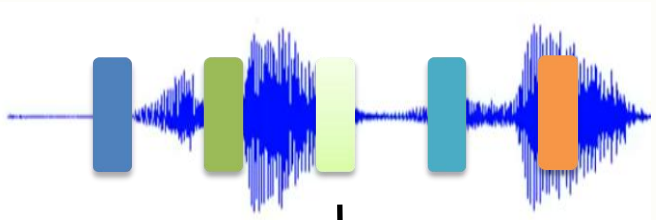


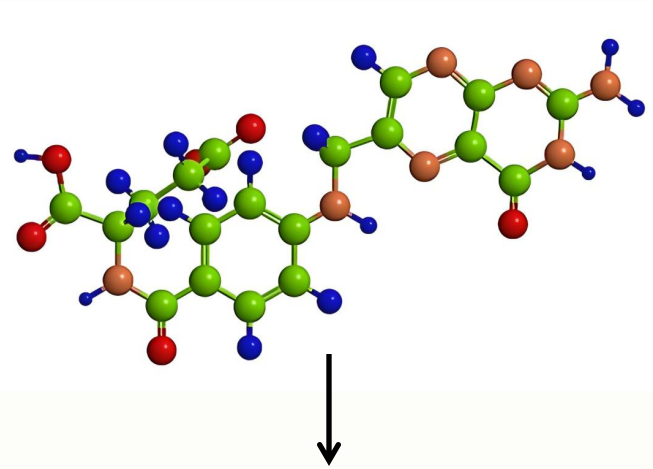
自注意力机制

多种多样的输入

- *整个序列输出一个结果（全局标签预测）*

Citizens have the right to right wrongs





中性语境

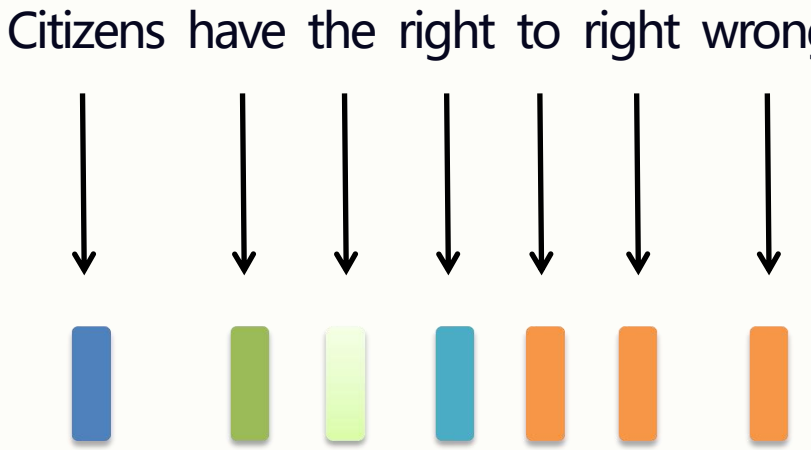
詹姆斯·邦德

弱毒性

自注意力机制

多种多样的输入

- ***模型自己决定输出的长度（seq2seq）***



→ 公民享有纠正错误的权利

→ 公民有权纠正不公

→ Les citoyens ont le droit de corriger les injustices.

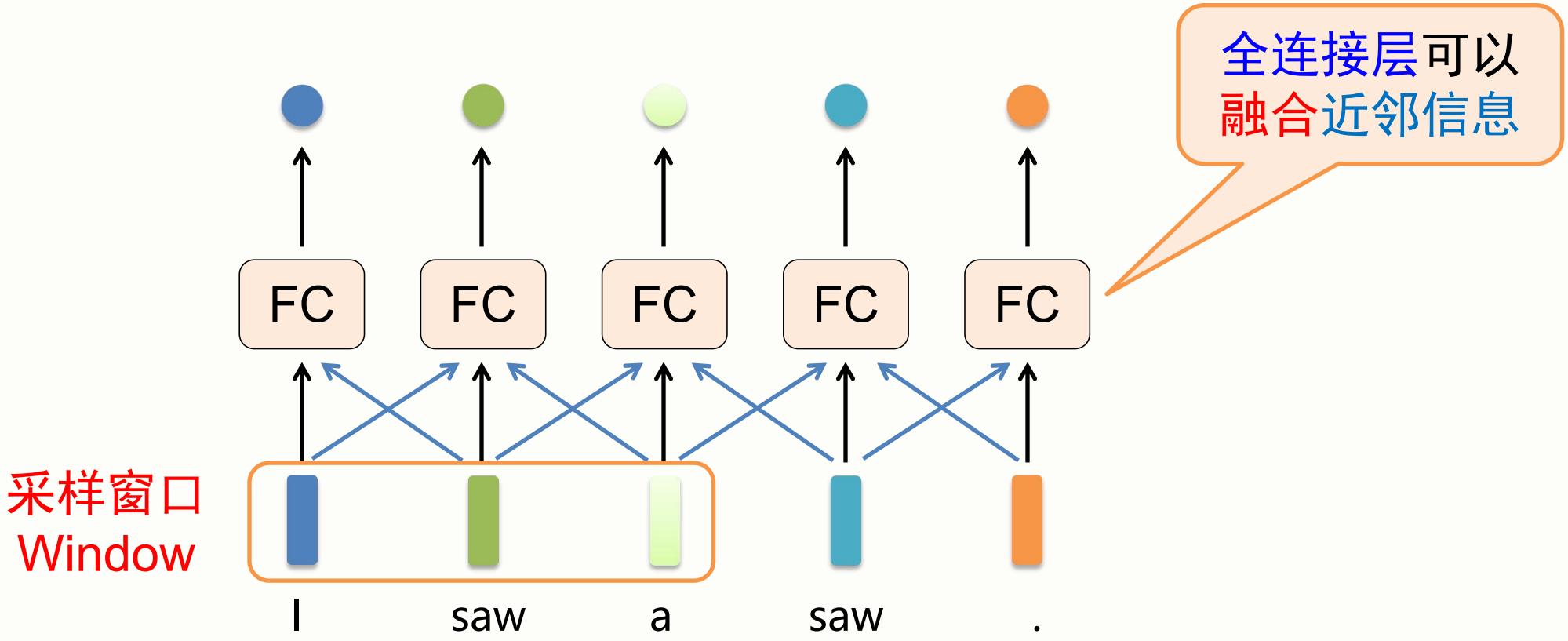
→ 市民は不公正を是正する権利を持っている。

自注意力（Self-Attention）

自注意力机制

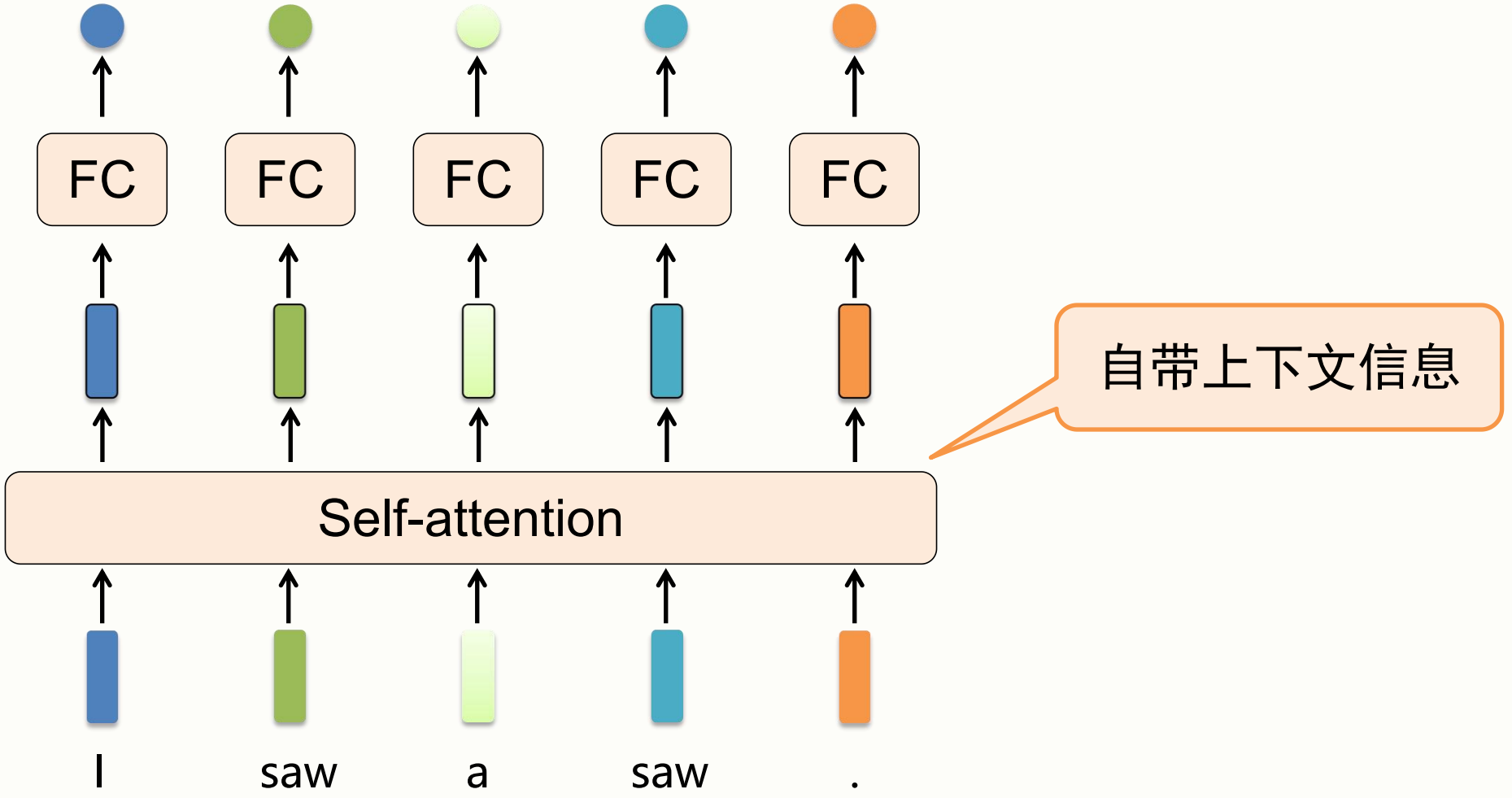
序列标注（Sequence Labeling）

- *每个向量输出一个结果*



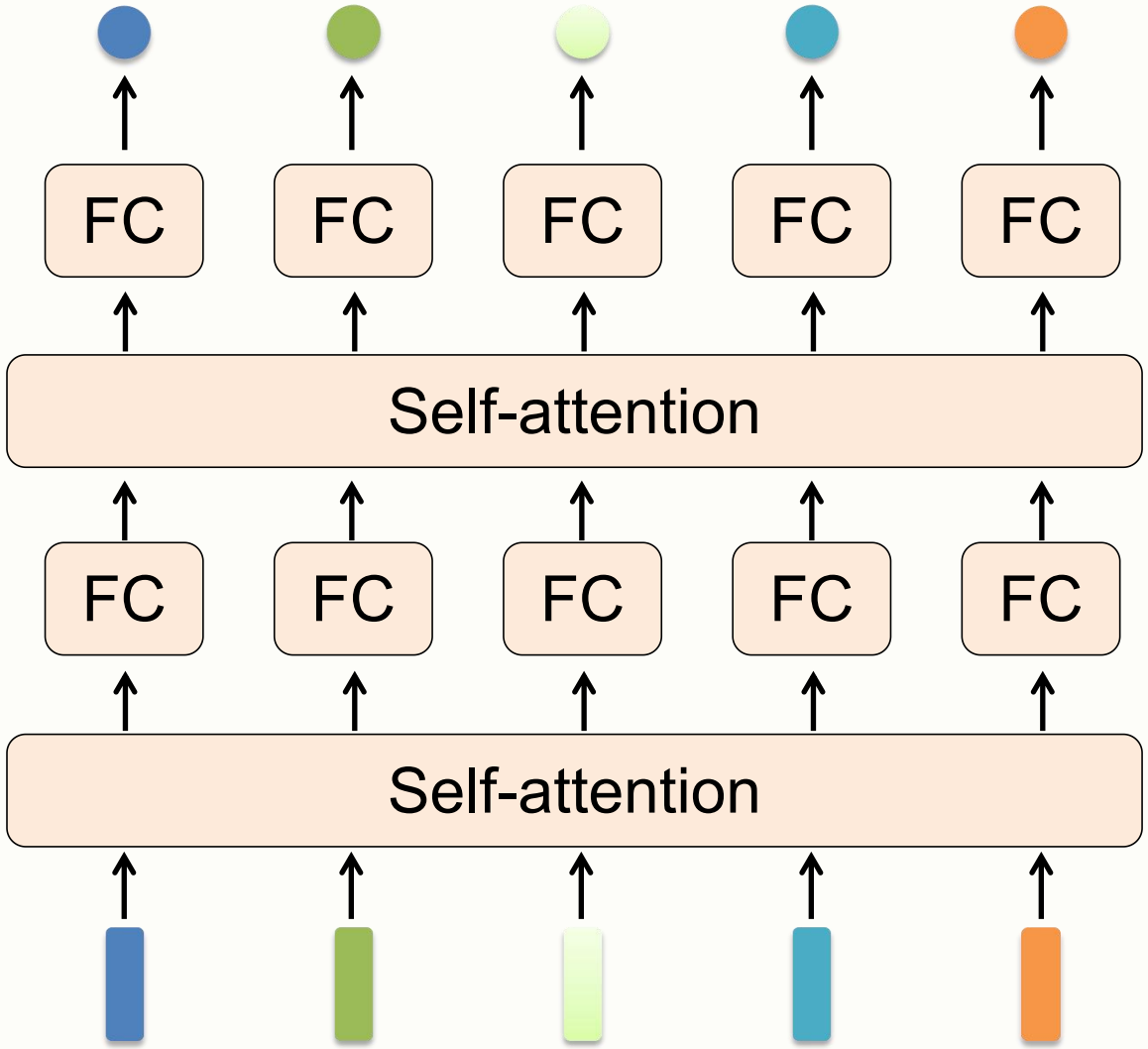
自注意力机制

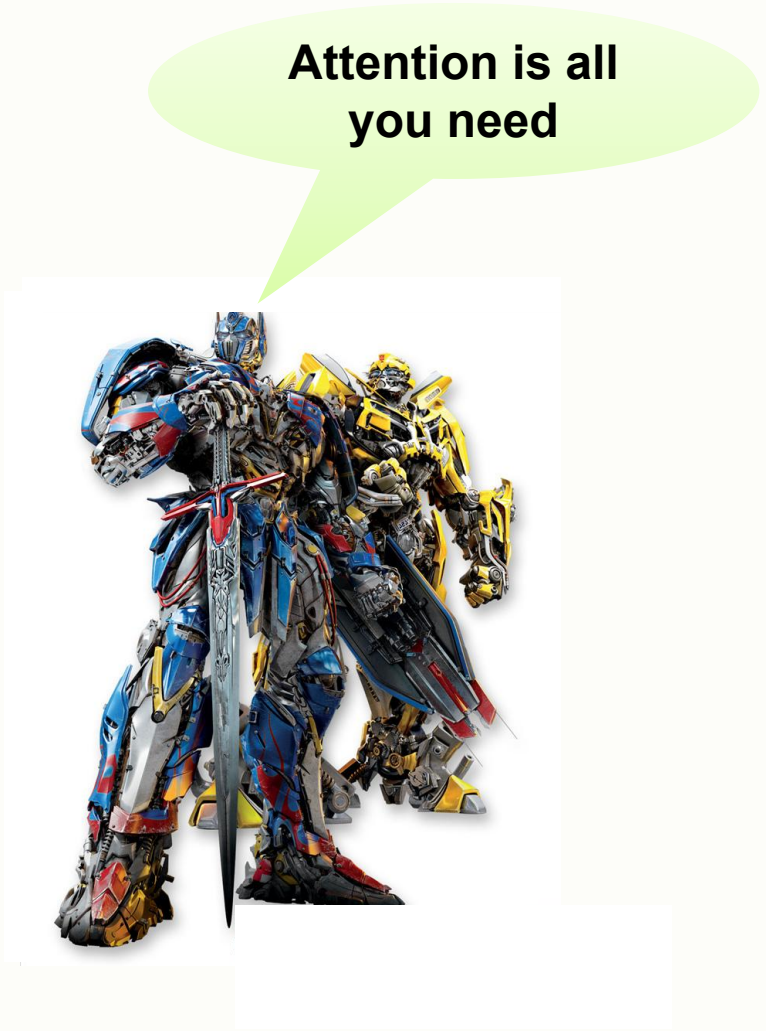
从序列标注到自注意力



自注意力机制

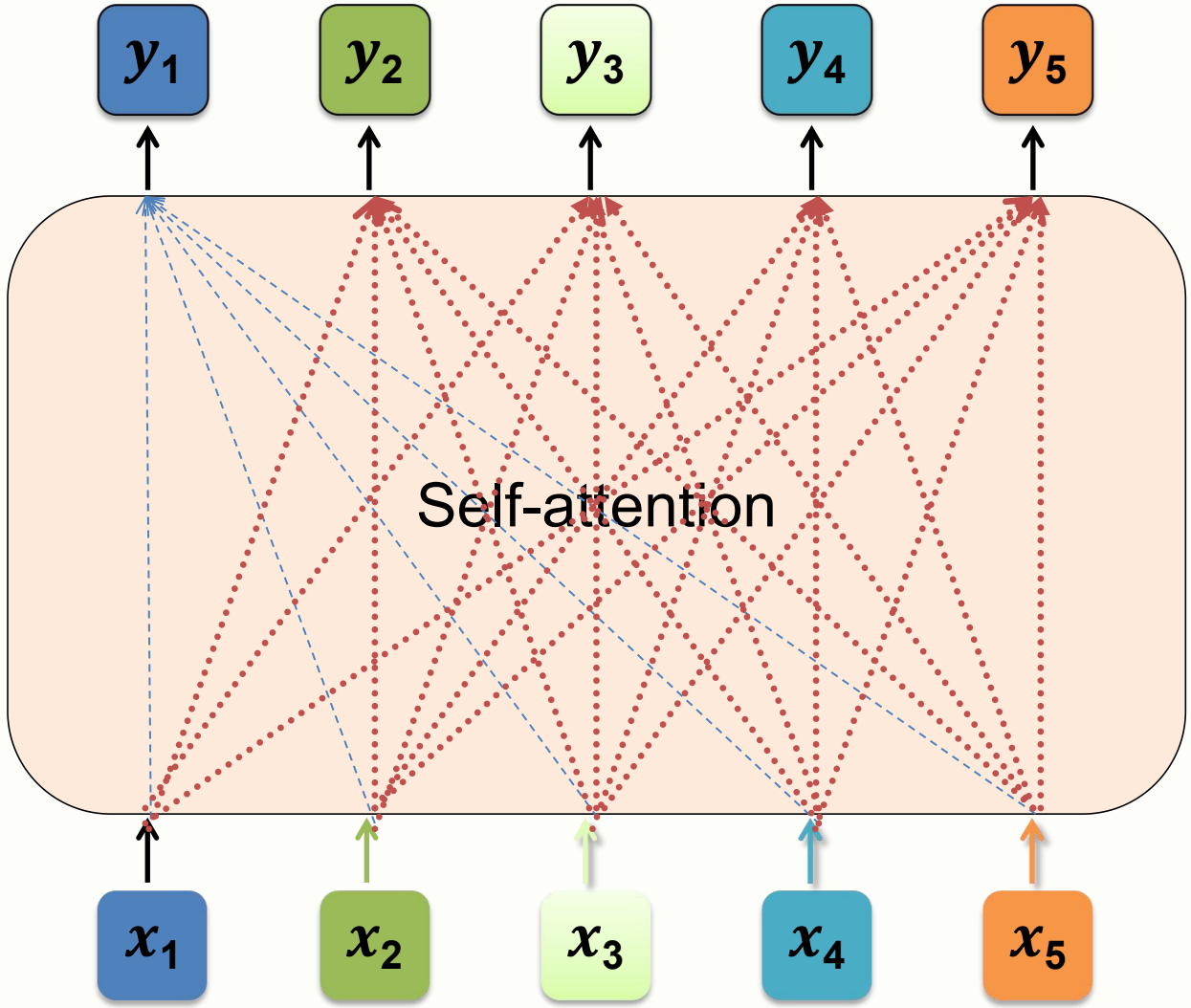
多级自注意力





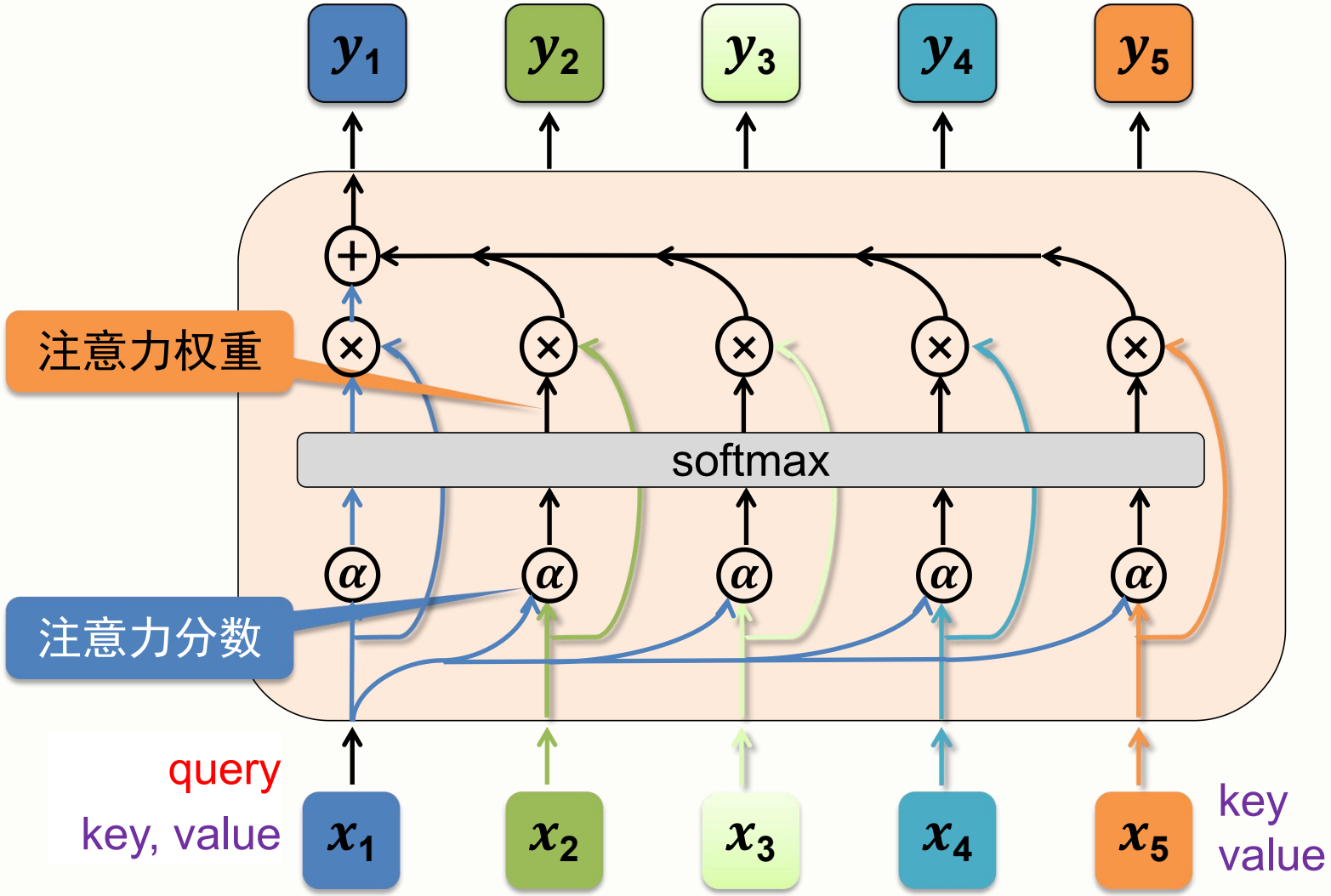
自注意力机制

自注意力的基本原理



自注意力机制

自注意力的基本原理



给定序列 $x_1, x_2, \ldots, x_n$, $\forall x_i \in \mathbb{R}^d$

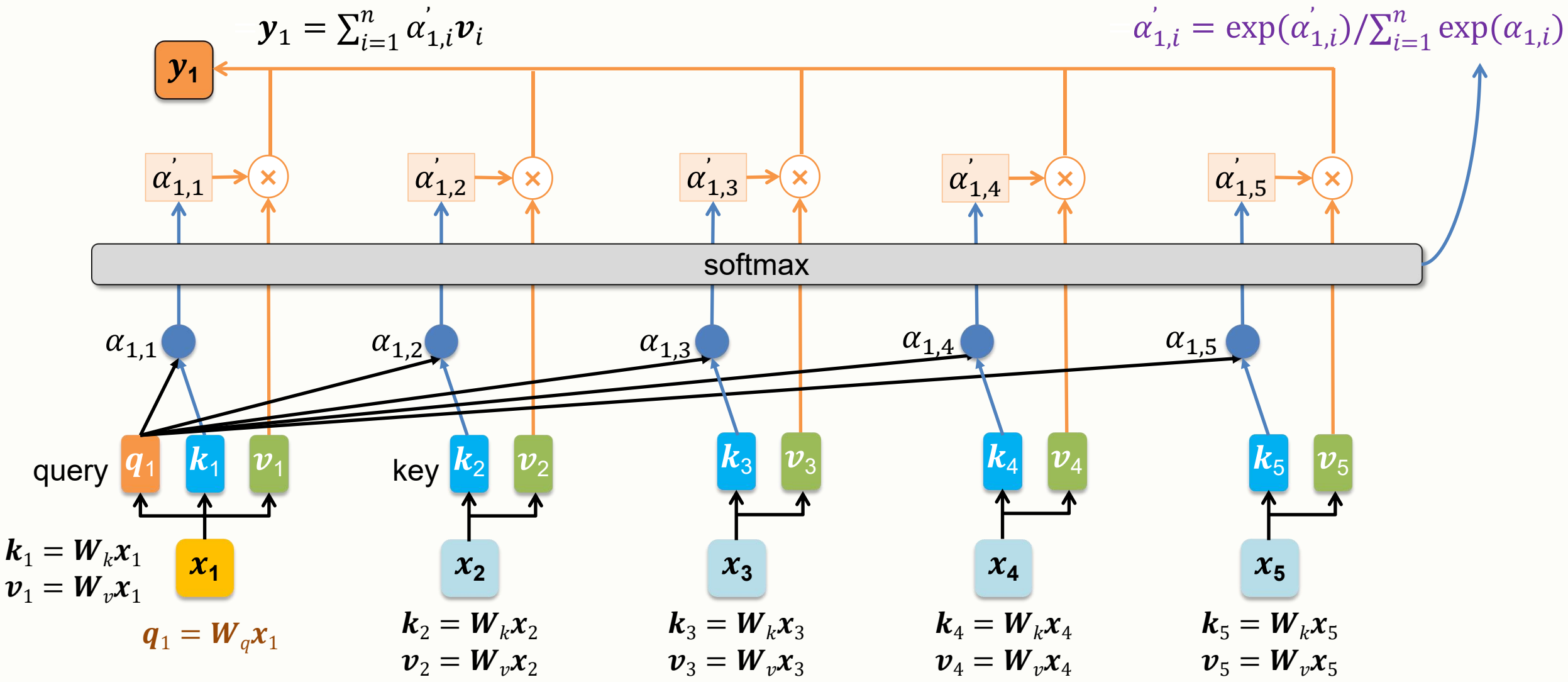
- ✓ 当 $i = 1$ 时，x_i 作为 query 输入到自注意力池化层
- ✓ 当 $i \neq 1$ 时，x_i 作为 key 和 value 输入到自注意力池化层

$$y_i = \text{softmax}(\frac{q\boldsymbol{K}^T}{\sqrt{d}})\boldsymbol{V}$$

$$= \text{softmax}(\frac{x_i\boldsymbol{X}^T}{\sqrt{d}})\boldsymbol{X}$$

自注意力机制

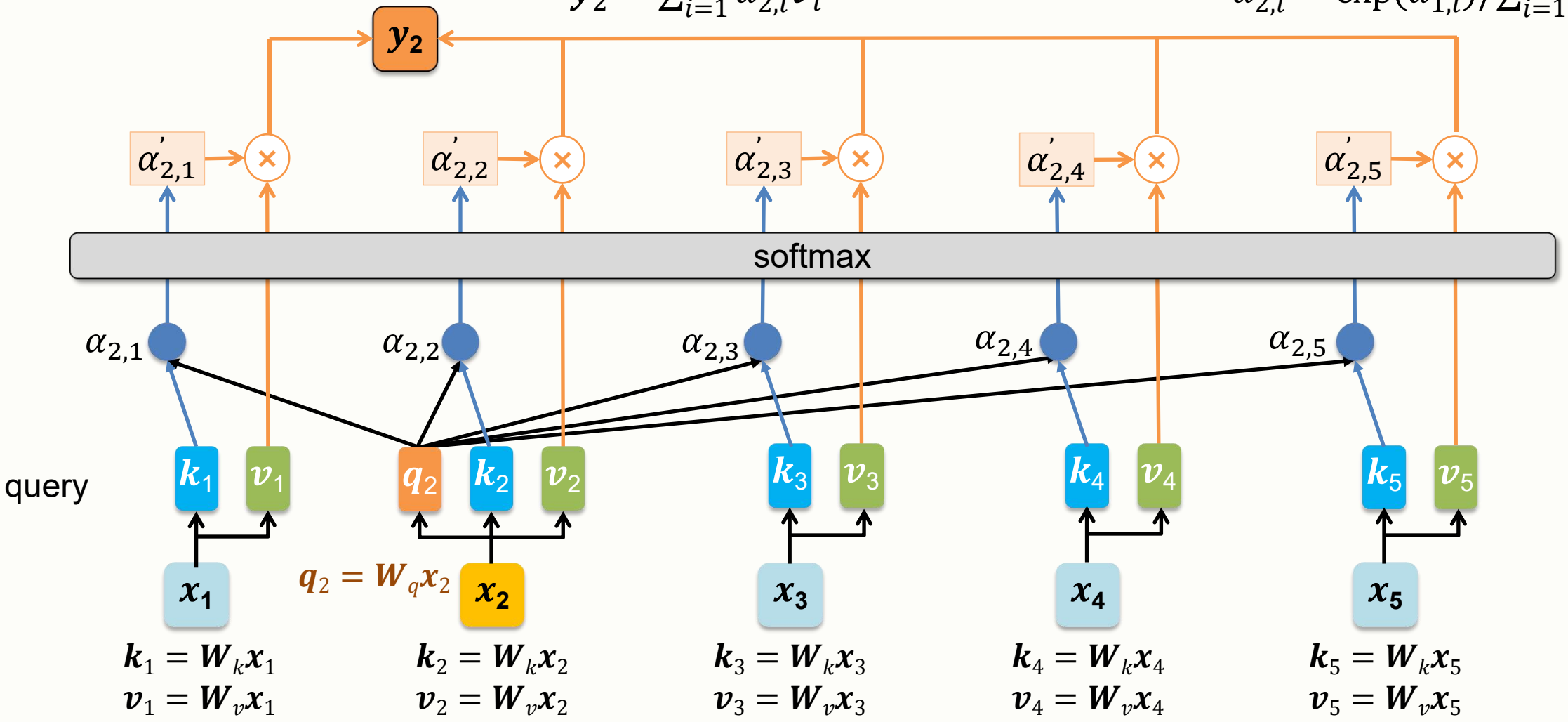
自注意力的数学表达



自注意力机制

自注意力的数学表达

$$y_2 = \sum_{i=1}^{n} \alpha'_{2,i} v_i \qquad \alpha'_{2,i} = \exp(\alpha'_{1,i}) / \sum_{i=1}^{n} \exp(\alpha_{1,i})$$



自注意力机制

自注意力的矩阵表达

$$q_i = W_q x_i$$

$$W_q \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{matrix} = \begin{matrix} q_1 & q_2 & q_3 & q_4 & q_5 \end{matrix}$$

X Q

$$k_i = W_k x_i$$

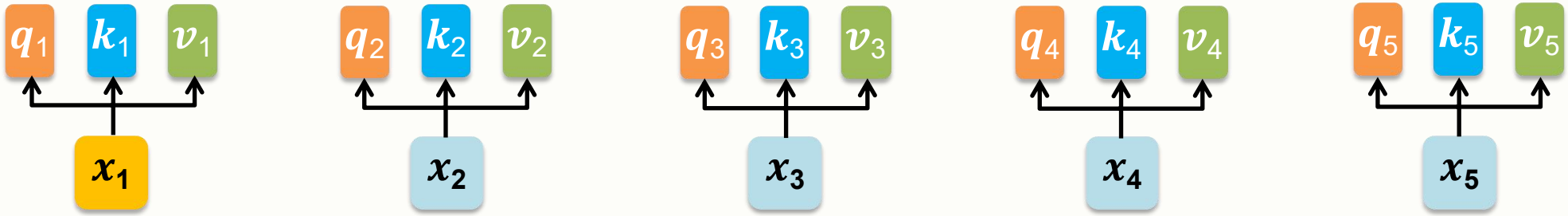
$$W_k \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{matrix} = \begin{matrix} k_1 & k_2 & k_3 & k_4 & k_5 \end{matrix}$$

X K

$$v_i = W_v x_i$$

$$W_v \begin{matrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{matrix} = \begin{matrix} v_1 & v_2 & v_3 & v_4 & v_5 \end{matrix}$$

X V



自注意力机制

自注意力的数学表达

$k_1 q_1 = \alpha_{1,1}$

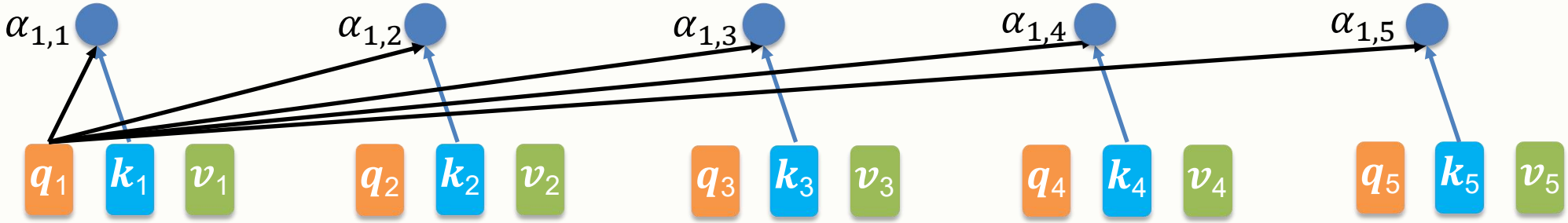
$k_2 q_1 = \alpha_{1,2}$

$k_3 q_1 = \alpha_{1,3}$

$k_4 q_1 = \alpha_{1,4}$

$k_5 q_1 = \alpha_{1,5}$

$$\begin{matrix} k_1 \\ k_2 \\ k_3 \\ k_4 \\ k_5 \end{matrix} q_1 = \begin{matrix} \alpha_{1,1} \\ \alpha_{1,2} \\ \alpha_{1,3} \\ \alpha_{1,4} \\ \alpha_{1,5} \end{matrix}$$



$$\begin{matrix} k_1 \\ k_2 \\ k_3 \\ k_4 \\ k_5 \end{matrix} \quad K^T$$

$$\begin{matrix} q_1 & q_2 & q_3 & q_4 & q_5 \end{matrix} \quad Q$$

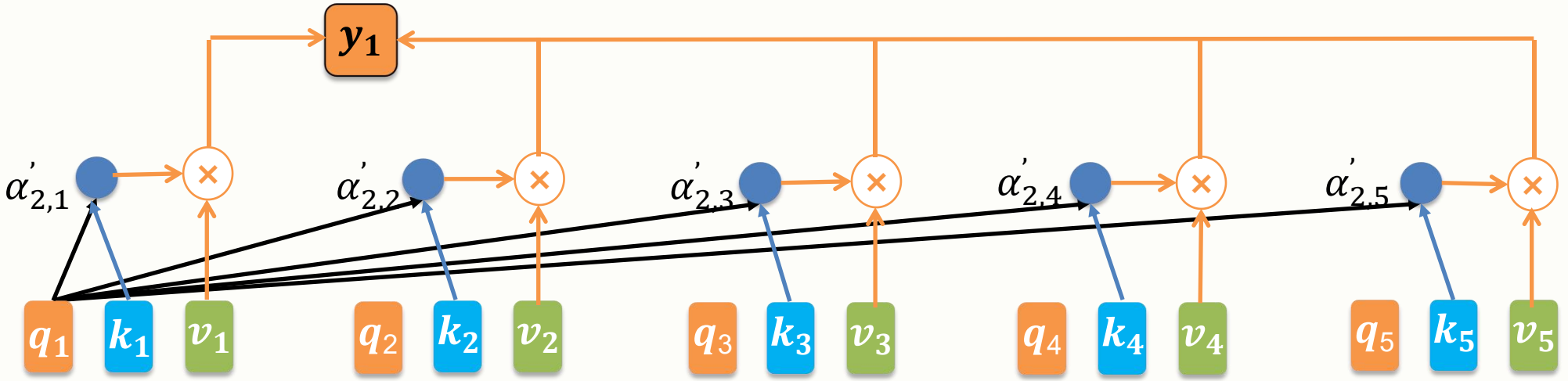
$$= \begin{matrix} \alpha_{1,1} & \alpha_{2,1} & \alpha_{3,1} & \alpha_{4,1} & \alpha_{5,1} \\ \alpha_{1,2} & \alpha_{2,2} & \alpha_{3,2} & \alpha_{4,2} & \alpha_{5,2} \\ \alpha_{1,3} & \alpha_{2,3} & \alpha_{3,3} & \alpha_{4,3} & \alpha_{5,3} \\ \alpha_{1,4} & \alpha_{2,4} & \alpha_{3,4} & \alpha_{4,4} & \alpha_{5,4} \\ \alpha_{1,5} & \alpha_{2,5} & \alpha_{3,5} & \alpha_{4,5} & \alpha_{5,5} \end{matrix} \quad A$$

softmax

$$\begin{matrix} \alpha'_{1,1} & \alpha'_{2,1} & \alpha'_{3,1} & \alpha'_{4,1} & \alpha'_{5,1} \\ \alpha'_{1,2} & \alpha'_{2,2} & \alpha'_{3,2} & \alpha'_{4,2} & \alpha'_{5,2} \\ \alpha'_{1,3} & \alpha'_{2,3} & \alpha'_{3,3} & \alpha'_{4,3} & \alpha'_{5,3} \\ \alpha'_{1,4} & \alpha'_{2,4} & \alpha'_{3,4} & \alpha'_{4,4} & \alpha'_{5,4} \\ \alpha'_{1,5} & \alpha'_{2,5} & \alpha'_{3,5} & \alpha'_{4,5} & \alpha'_{5,5} \end{matrix} \quad A'$$

自注意力机制

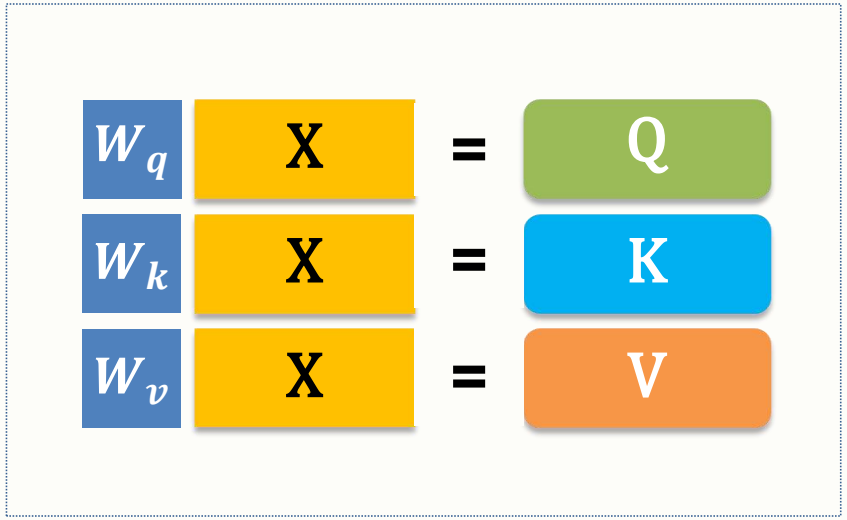
自注意力的数学表达

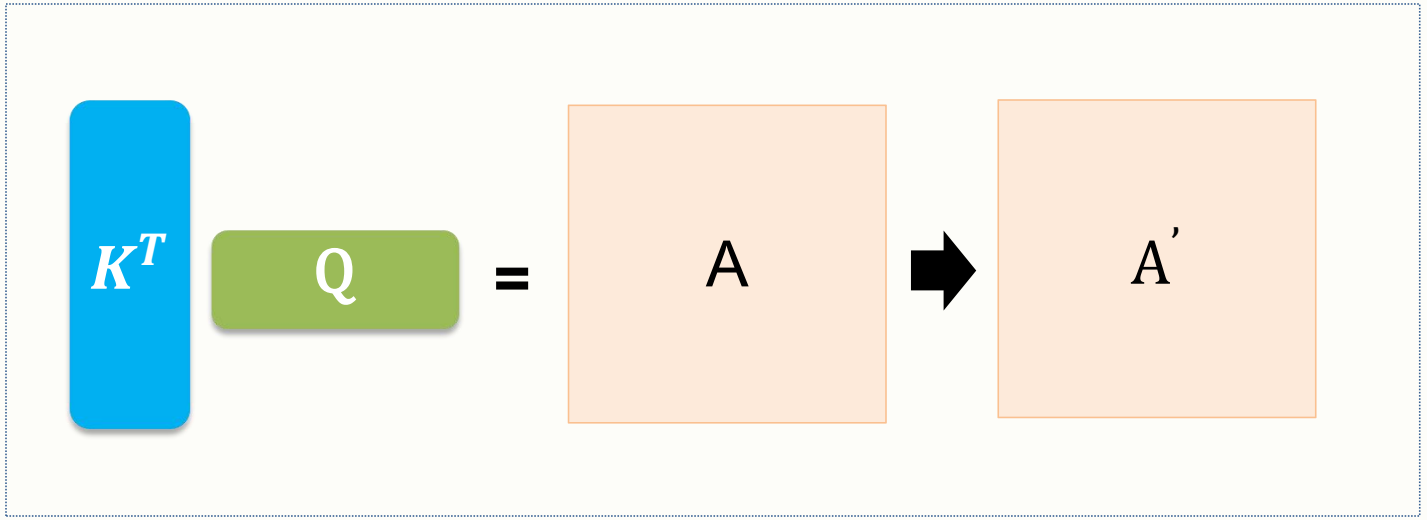


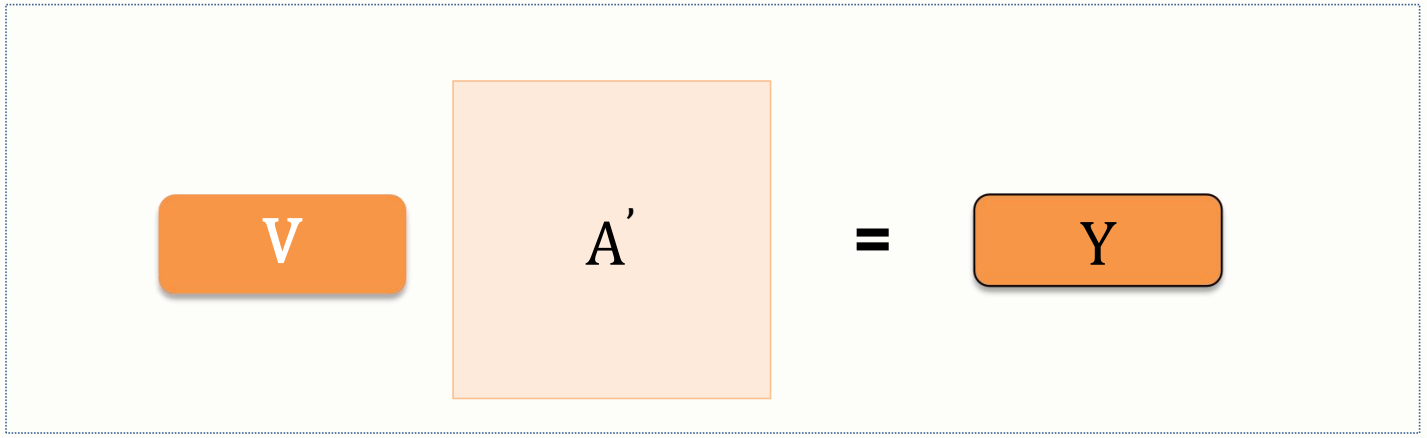
$$\underbrace{\begin{bmatrix} y_1 & y_2 & y_3 & y_4 & y_5 \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} v_1 & v_2 & v_3 & v_4 & v_5 \end{bmatrix}}_{V} \underbrace{\begin{bmatrix} \alpha'_{1,1} & \alpha'_{2,1} & \alpha'_{3,1} & \alpha'_{4,1} & \alpha'_{5,1} \\ \alpha'_{1,2} & \alpha'_{2,2} & \alpha'_{3,2} & \alpha'_{4,2} & \alpha'_{5,2} \\ \alpha'_{1,3} & \alpha'_{2,3} & \alpha'_{3,3} & \alpha'_{4,3} & \alpha'_{5,3} \\ \alpha'_{1,4} & \alpha'_{2,4} & \alpha'_{3,4} & \alpha'_{4,4} & \alpha'_{5,4} \\ \alpha'_{1,5} & \alpha'_{2,5} & \alpha'_{3,5} & \alpha'_{4,5} & \alpha'_{5,5} \end{bmatrix}}_{A'}$$

自注意力机制

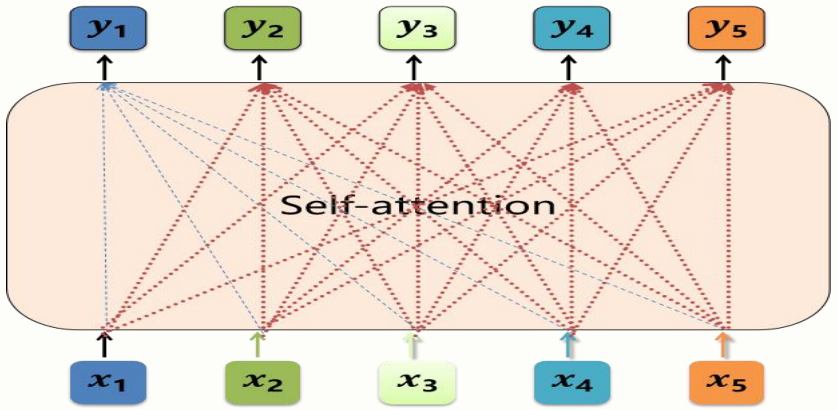
自注意力的数学表达





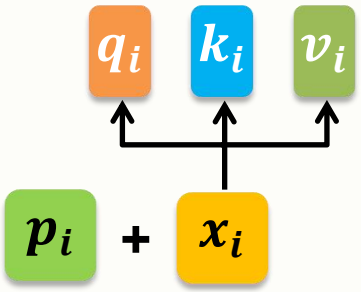


位置编码（position encoding）



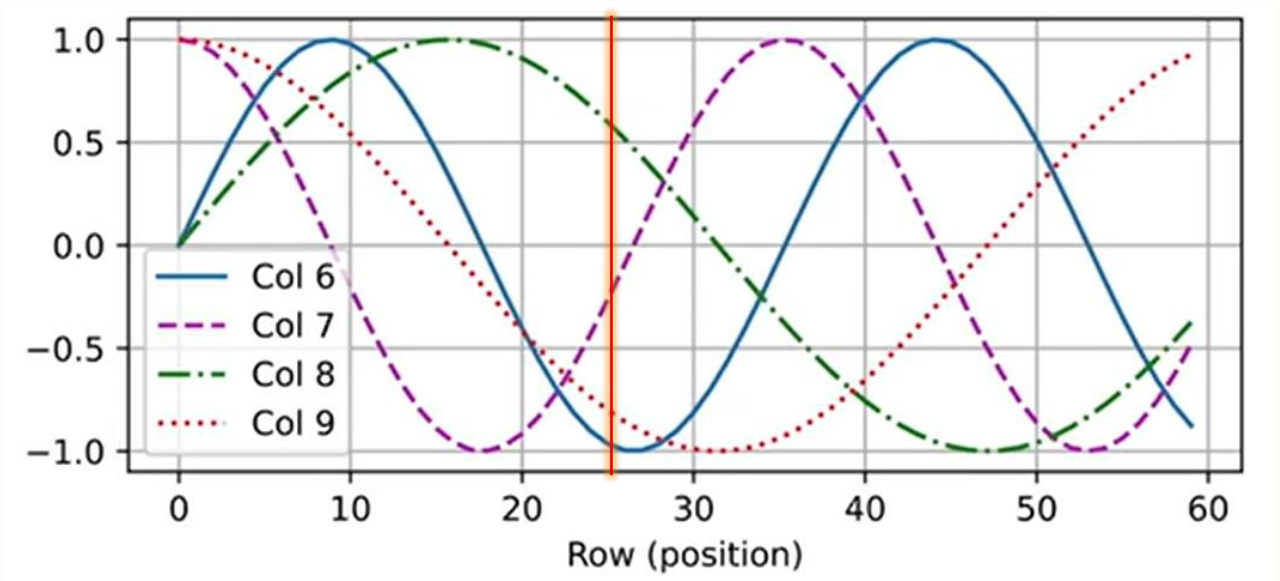
位置编码（position encoding）

位置编码矩阵



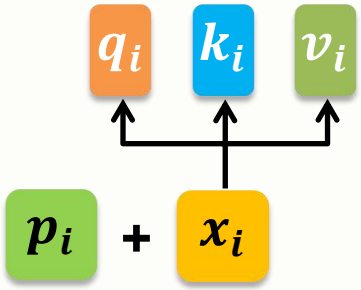
- 假设 $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ 是长为 n 的序列，则可以使用位置编码矩阵 $\boldsymbol{P} \in \mathbb{R}^{n \times d}$ 来将位置信息 $\boldsymbol{p_i}$ 手动注入到输入 $\boldsymbol{X}$ 中。$\boldsymbol{P}$ 的计算公式如下：

$$p_{i,2j} = sin(\frac{i}{10000^{2j/d}}), \qquad p_{i,2j+1} = cos(\frac{i}{10000^{2j/d}})$$



位置编码（position encoding）

位置编码的相对性



- 已知位置 i 处的位置编码，可以通过一个投影矩阵来获取 $i+\delta$ 处的位置编码。
- $i+\delta$ 处的位置编码与它在序列中的实际位置无关。
- 设 $\omega_j = 10000^{2j/d}$，则：

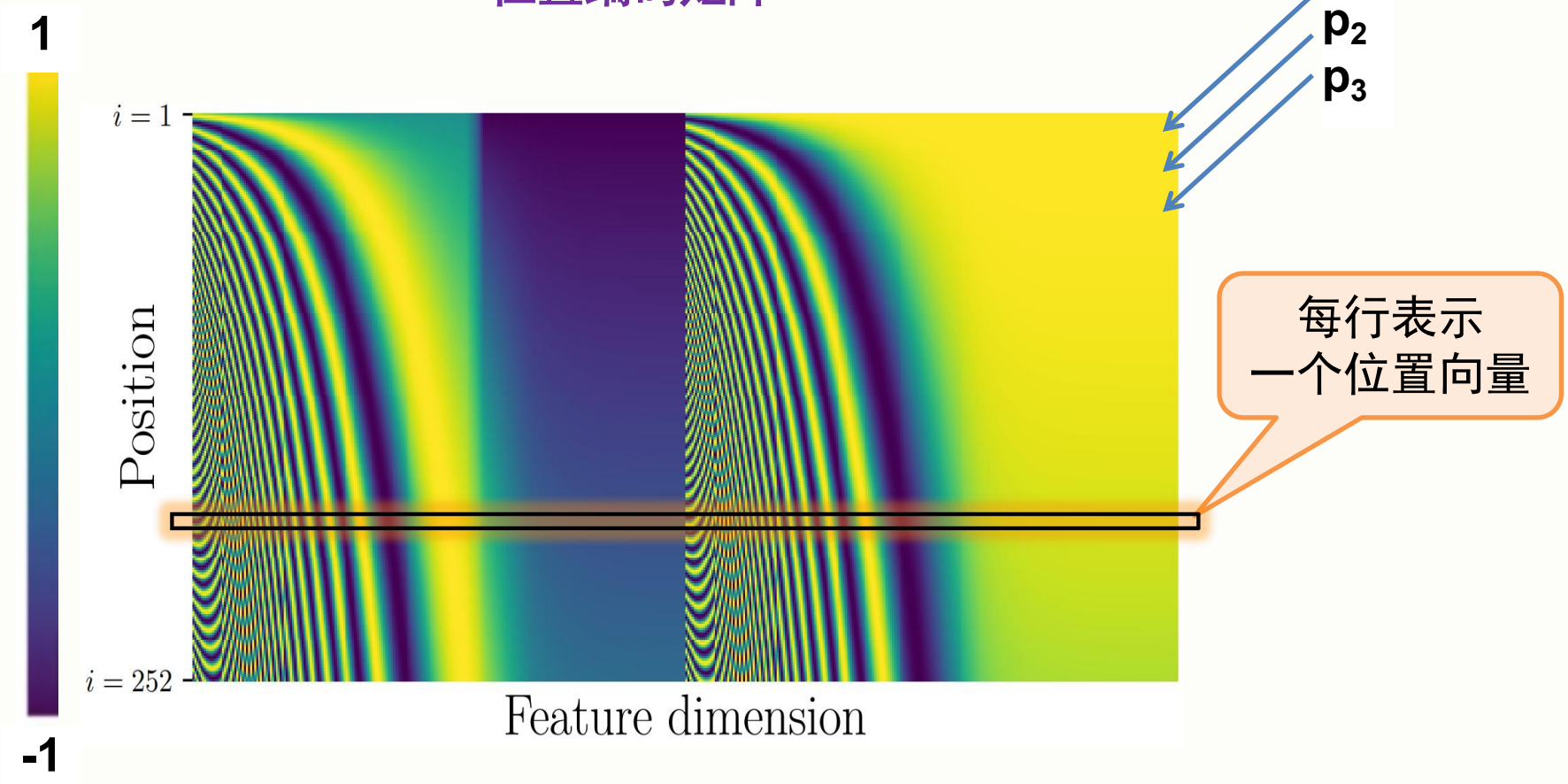
$$\begin{bmatrix} p_{i+\delta,2j} \\ p_{i+\delta,2j+1} \end{bmatrix} = \begin{bmatrix} cos(\delta\omega_j) & sin(\delta\omega_j) \\ -sin(\delta\omega_j) & cos(\delta\omega_j) \end{bmatrix} \begin{bmatrix} p_{i,2j} \\ p_{i,2j+1} \end{bmatrix}$$

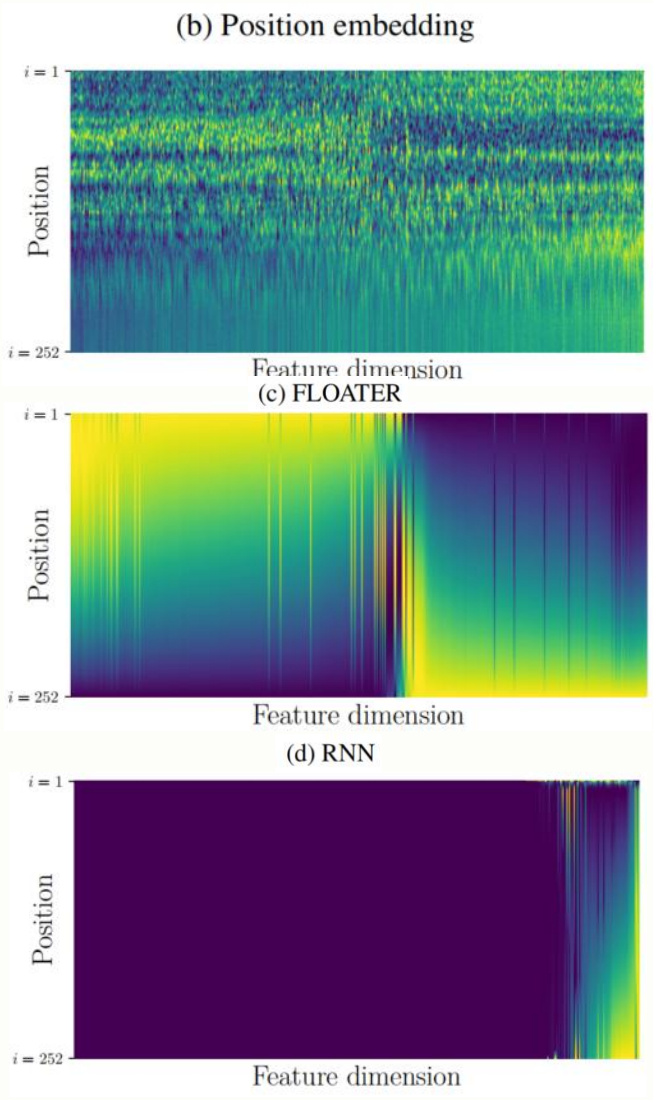
投影矩阵

位置编码（position encoding）

绝对位置信息

位置编码矩阵

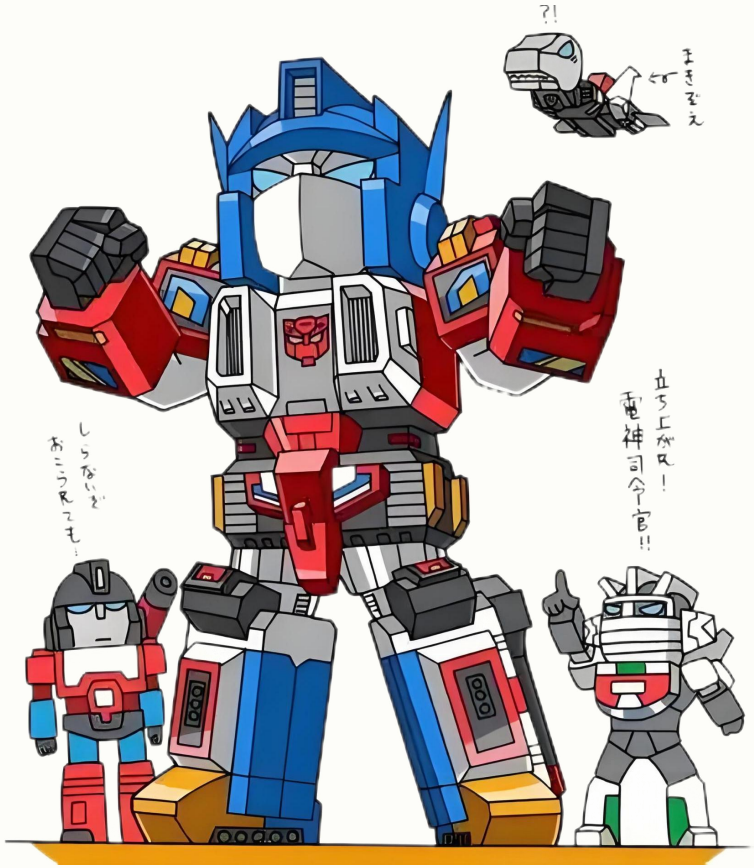




https://arxiv.org/abs/2003.09229

自注意力机制的应用

自注意力机制的应用



Transformer

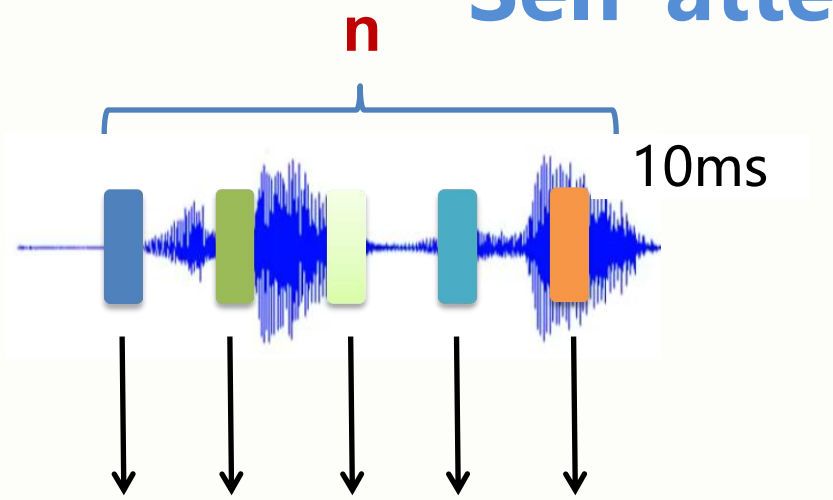
http://arxiv.org/abs/1706.03762

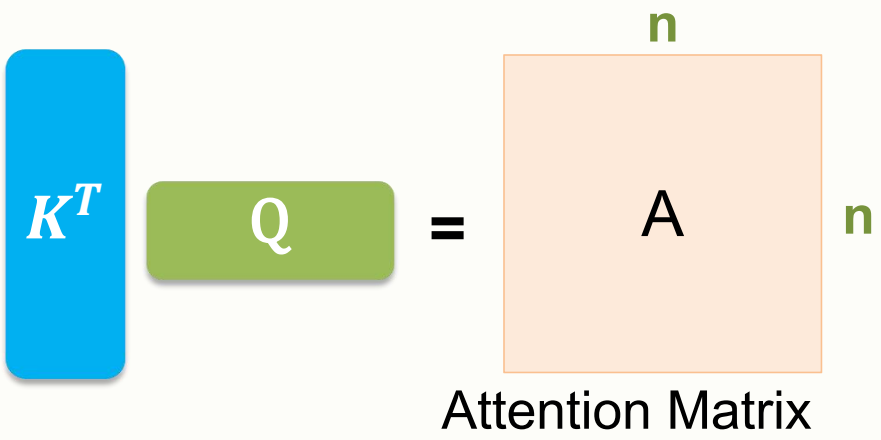
BERT

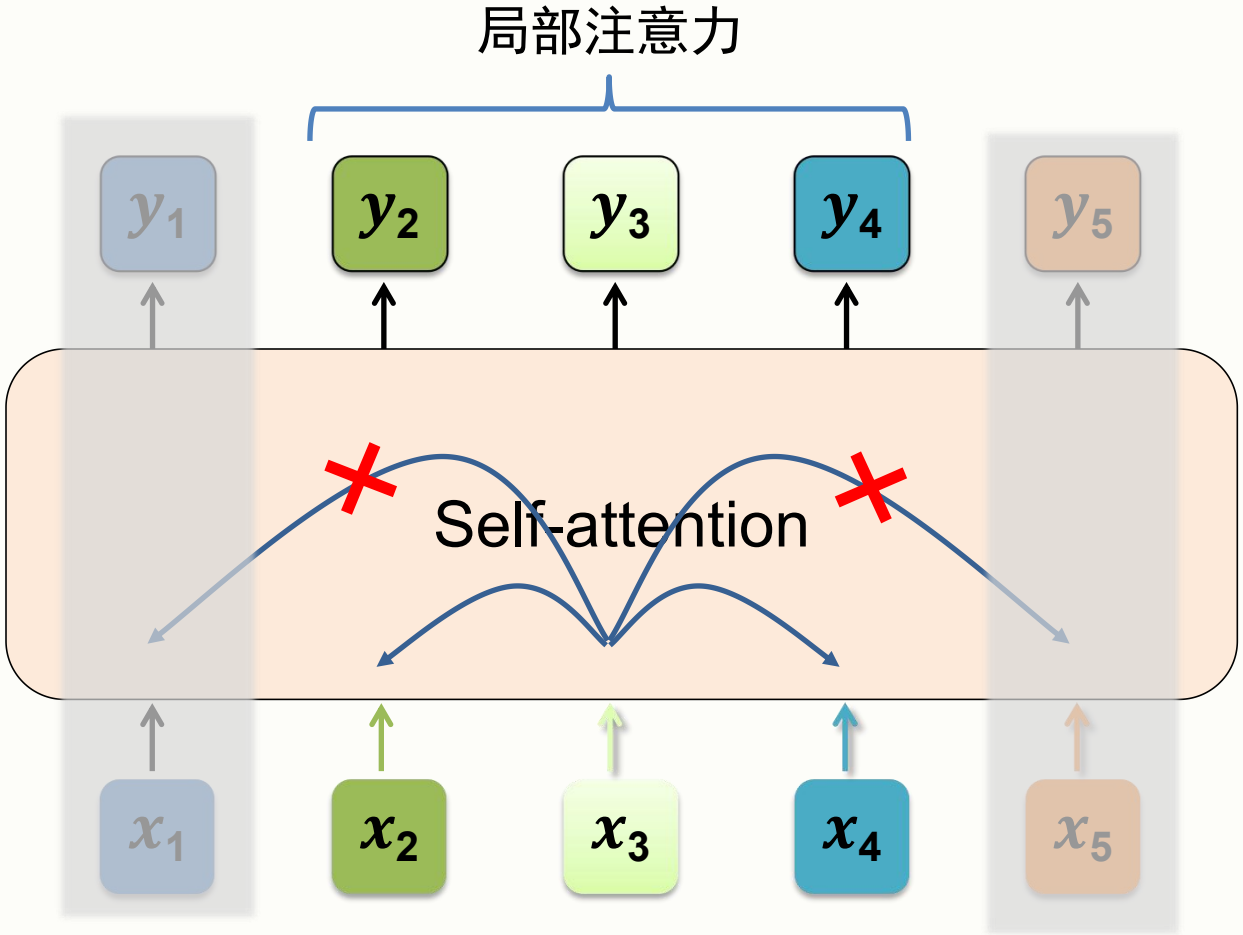
http://arxiv.org/abs/1810.04805

自注意力机制的应用

Self-attention for Speech



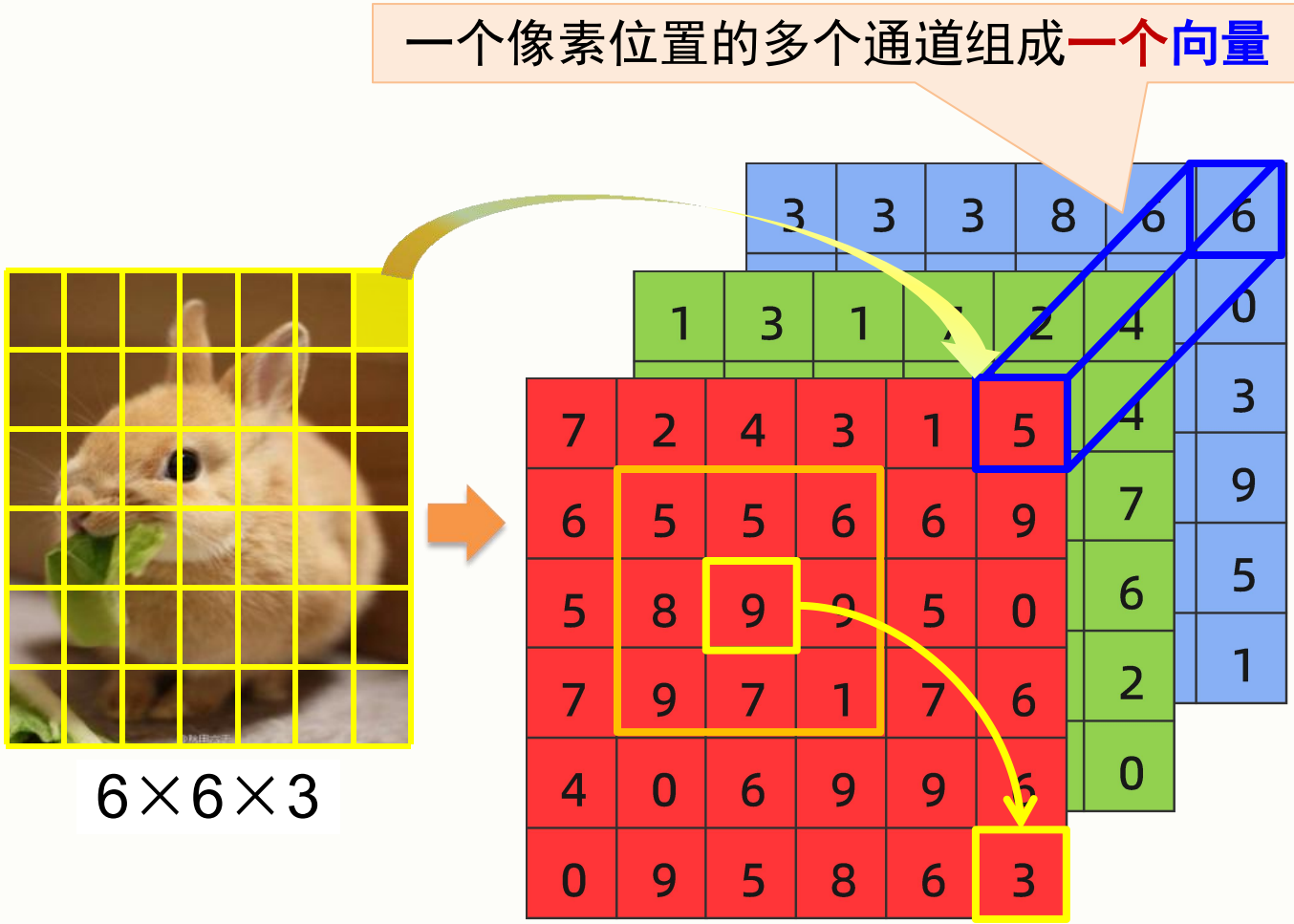




Truncated self-attetnion

自注意力机制的应用

Self-attention for Image



- CNN可以被看作是一个 receptive field 中的局部 self-attention。
  CNN是一个简化版的 self-attention
- Self-attention也可以被看作是一个包含可学习参数的CNN。
  Self-attention是一个复杂版的CNN

[arXiv1911] On the Relationship between Self-Attention and Convolutional Layers

自注意力机制的应用

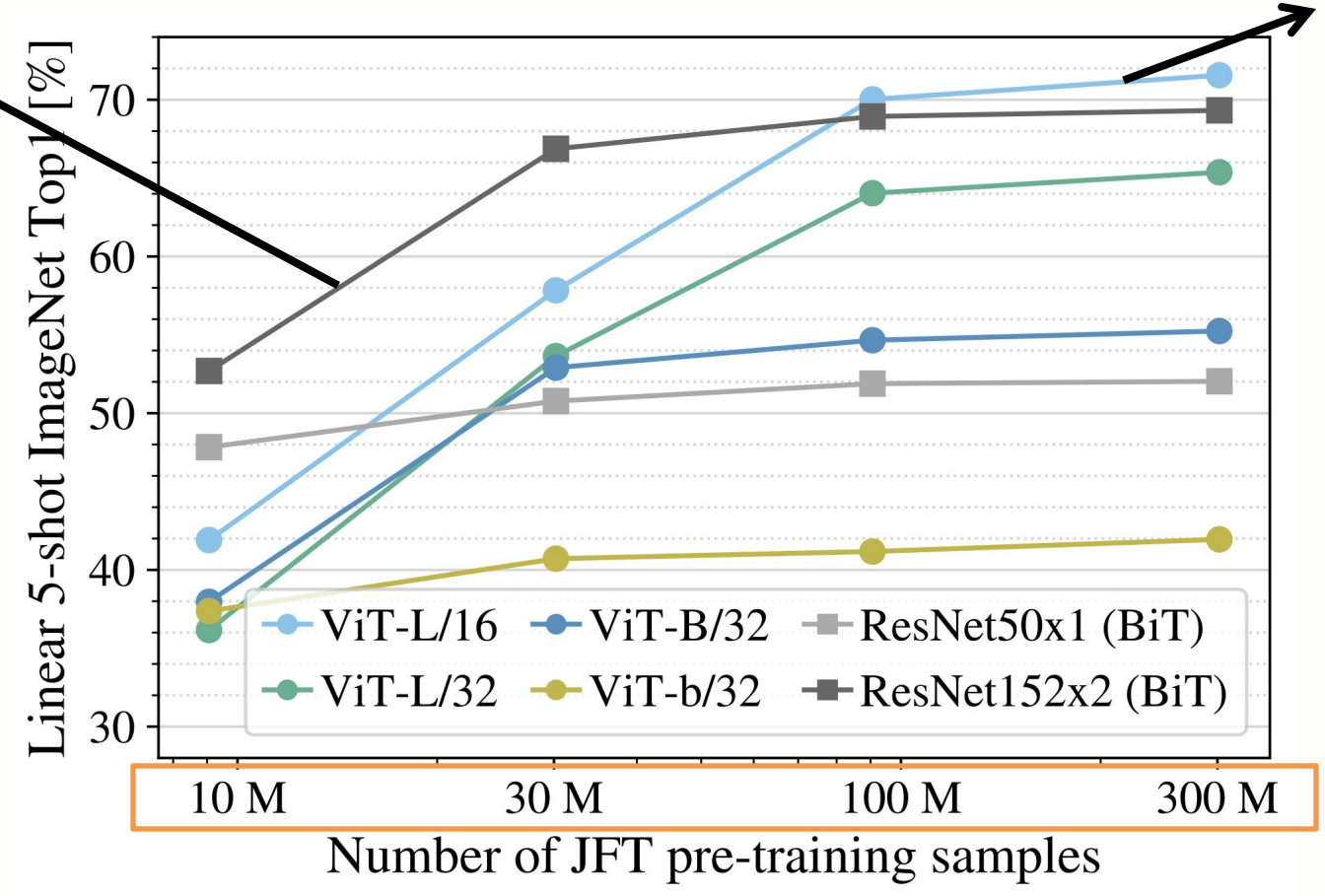
Self-attention for Image

CNN

数据较少时也有较好的性能

Self-attention

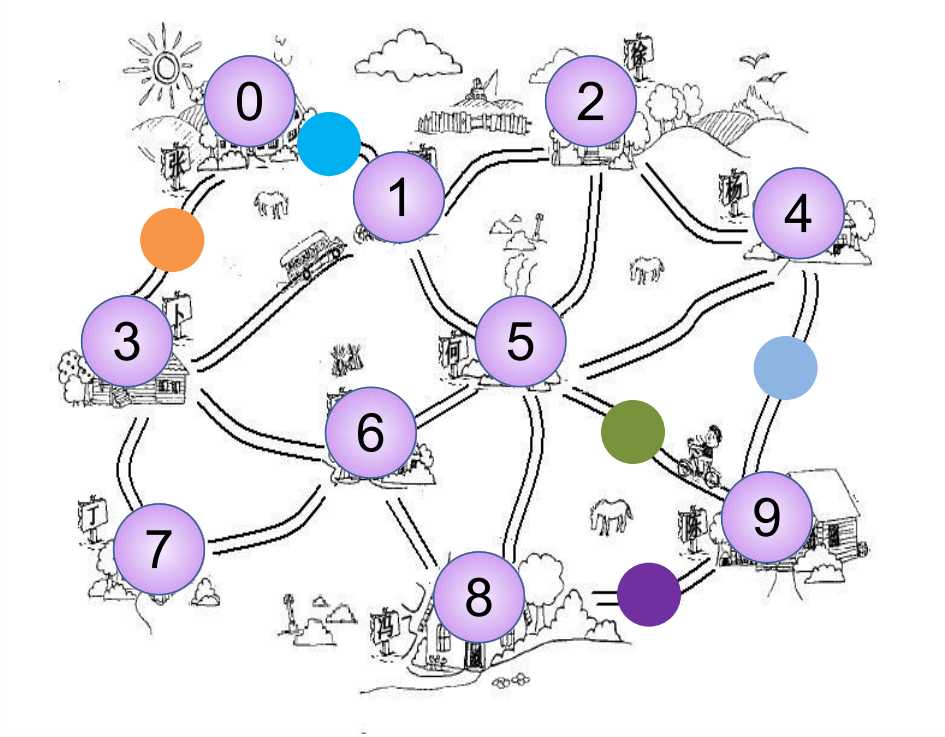
较多的数据能带来性能的提升



[arXiv2010] An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale

自注意力机制的应用

Self-attention for Graph



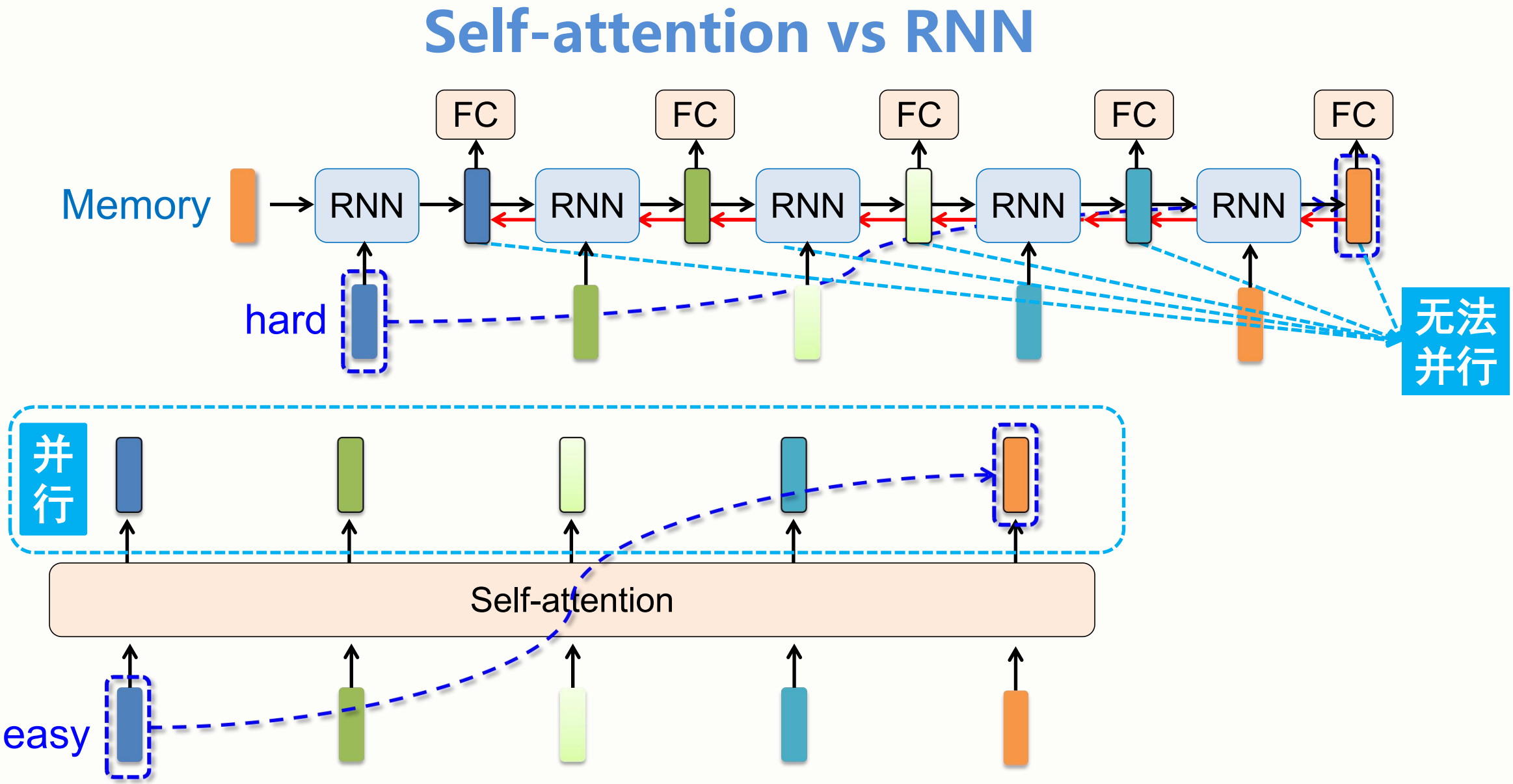
邻接结点，既包含结点，也包含边

邻接矩阵 *Attention Matrix*

	v_0	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_0	0	1	0	1	0	0	0	0	0	0
v_1	1	0	1	1	0	1	0	0	0	0
v_2	0	1	0	0	1	1	0	0	0	0
v_3	1	1	0	0	0	0	1	1	0	0
v_4	0	0	1	0	0	1	0	0	0	1
v_5	0	1	1	0	1	0	1	0	1	1
v_6	0	0	0	1	0	1	0	1	1	0
v_7	0	0	0	1	0	0	1	0	0	0
v_8	0	0	0	0	0	1	1	0	0	1
v_9	0	0	0	0	1	1	0	0	1	0

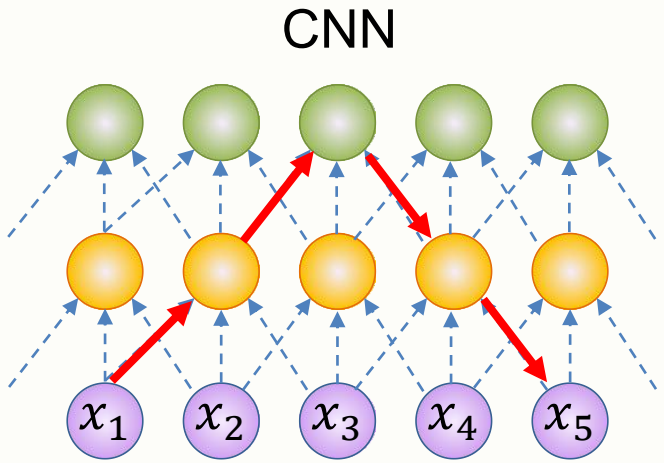
Graph Neural Network（GNN）

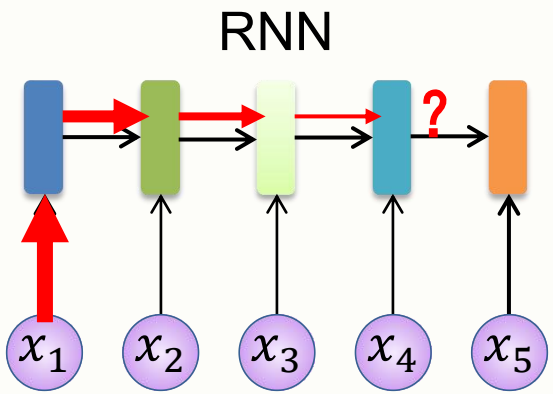
自注意力机制的应用

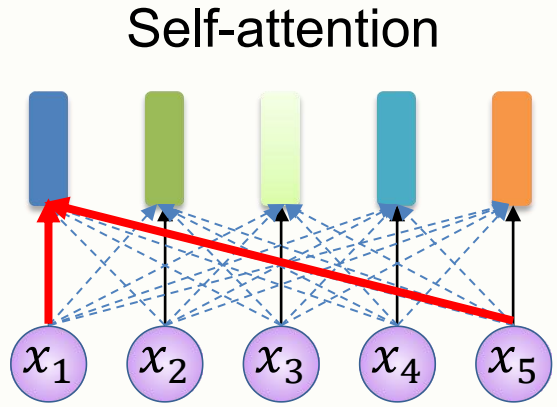

Self-attention vs RNN
FC
FC
FC
FC
FC
Memory
RNN
RNN
RNN
RNN
RNN
hard
无法并行
并行
Self-attention
easy

自注意力机制的应用

Self-attention vs CNN、RNN





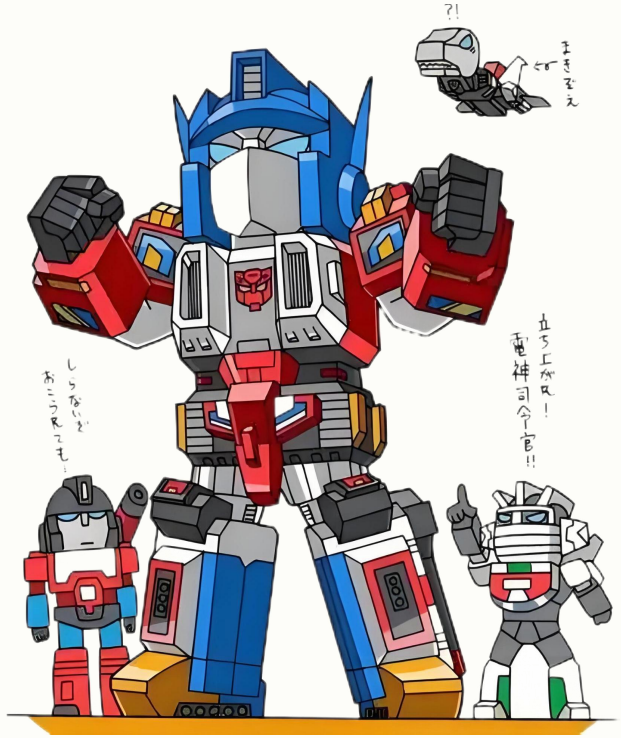


	CNN	RNN	自注意力
计算复杂度	$O(knd^2)$	$O(nd^2)$	$O(dn^2)$
并行度	$O(n)$	$O(1)$	$O(n)$
最长路径	$O\ (n/k)$	$O\ (n)$	$O\ (1)$

自注意力机制的应用

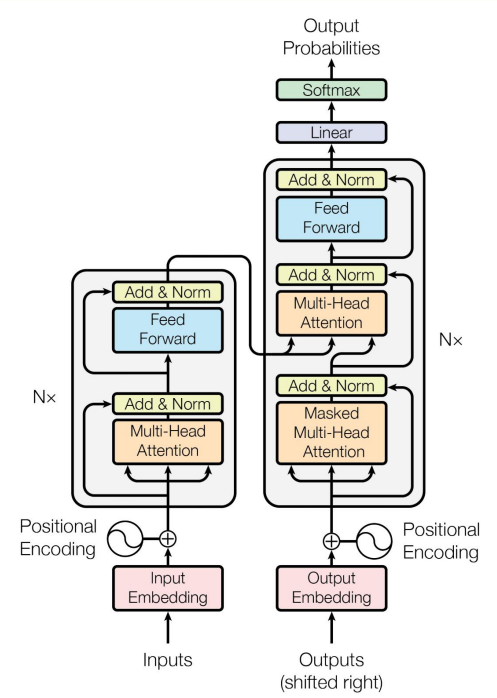
小　　结

- **自注意力机制的三重角色映射**
  - 输入向量 x_1 同时当作 query，key 和 value 来对序列抽取特征
  - 直接基于序列生成特征表示，无需额外信息注入
- **全局感知与计算效率的权衡**
  - 每个输出向量都具有完整序列上下文的感知能力
  - 完全并行化计算架构实现高效训练
  - 平方级计算复杂度($O(n^2)$)成为长序列处理的主要瓶颈
- **位置编码的时序注入机制**
  - 将位置信息直接注入输入向量，确保自注意力的固有排列不变性
  - Transformer 采用的 sin-cos 位置编码




TRANSFORMERS
THE LAST KNIGHT

- Transformer 的架构概述
- Transformer 的模块
- Transformer 的训练和预测

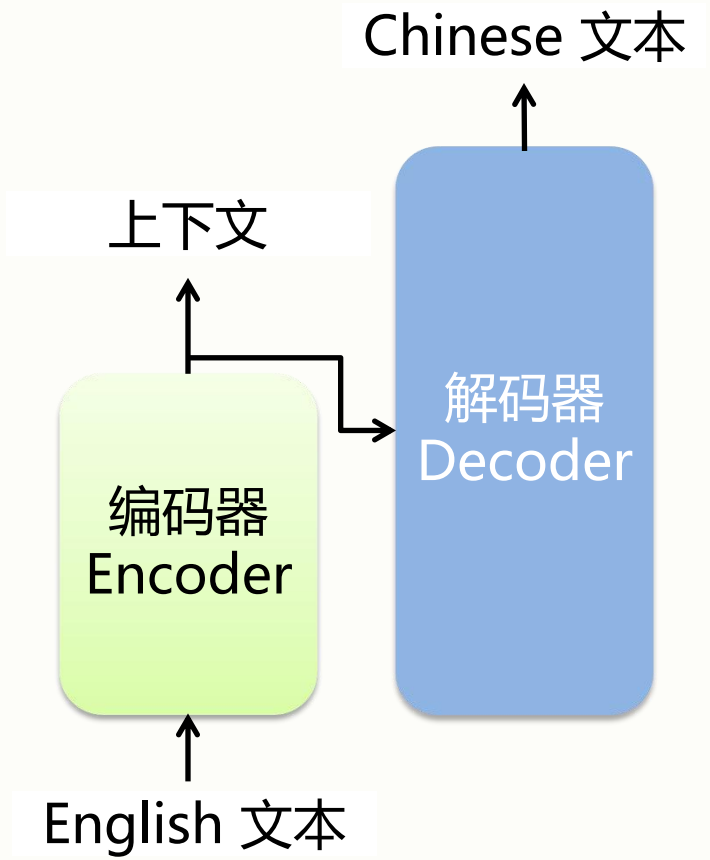


Transformer的架构概述

Ahish Vaswani, Noam Shazeer, Niki Parmar, et.al. Attention is all you need. NIPS2017.

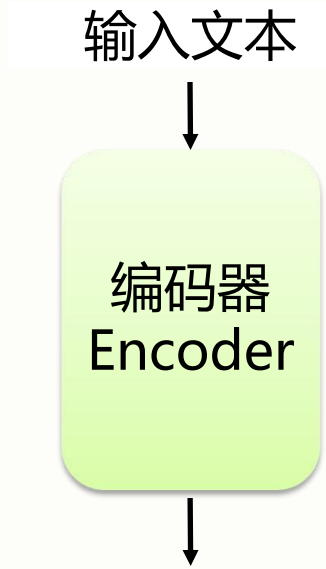
Transformer的架构概述

原始 Seq2Seq



将英文翻译成中文

编码器模型



Basis：

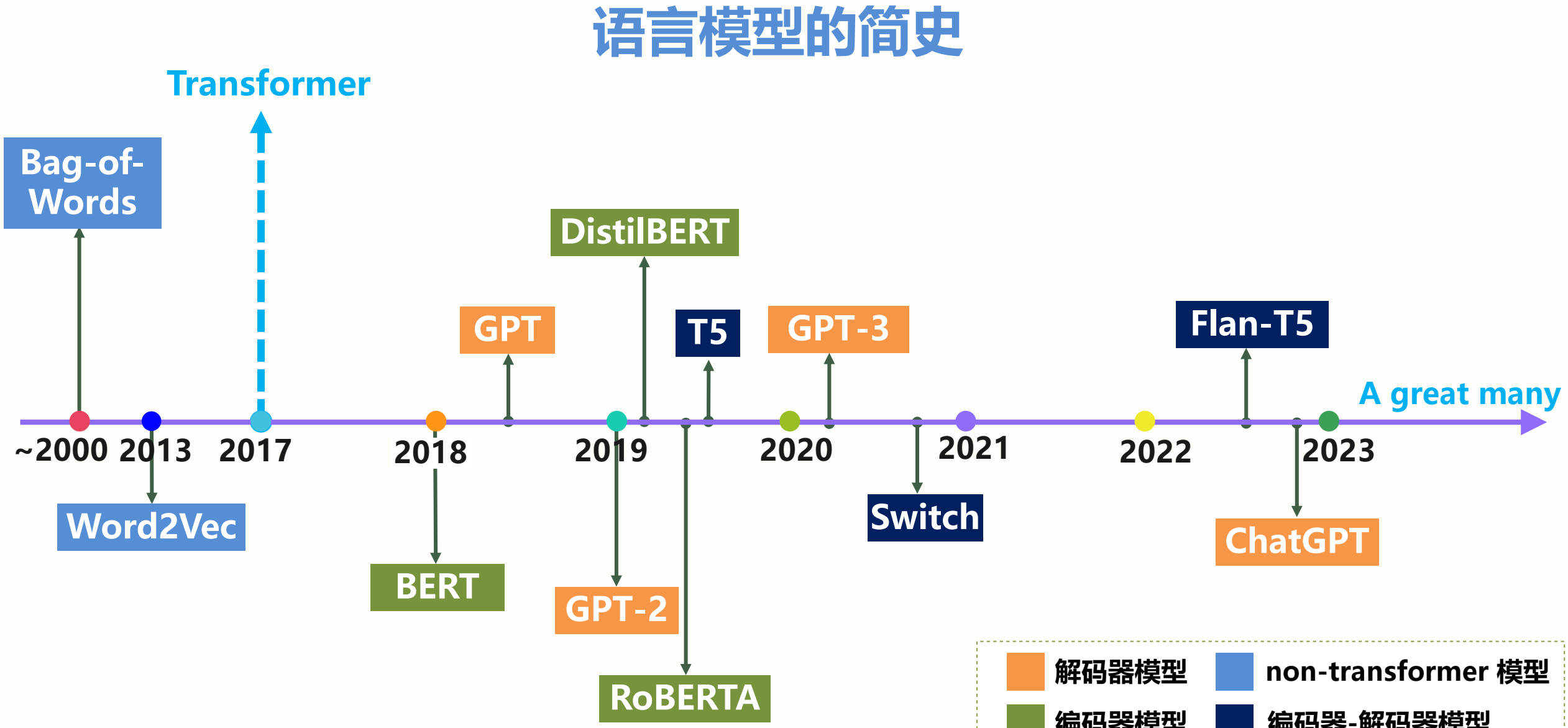
- BERT
- 大多数 embedding 模型

解码器模型

Basis：

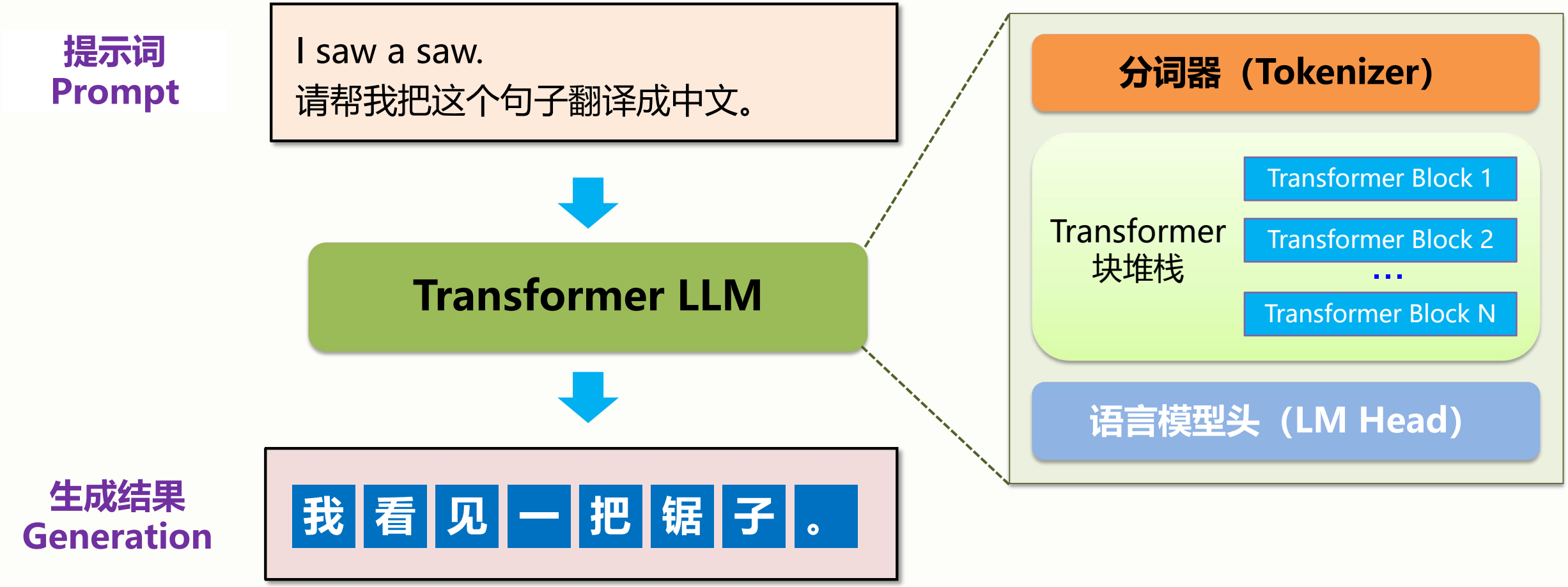
- 主流LLMs模型
- GPT、Claude、文心等

Transformer的架构概述



Transformer的架构概述

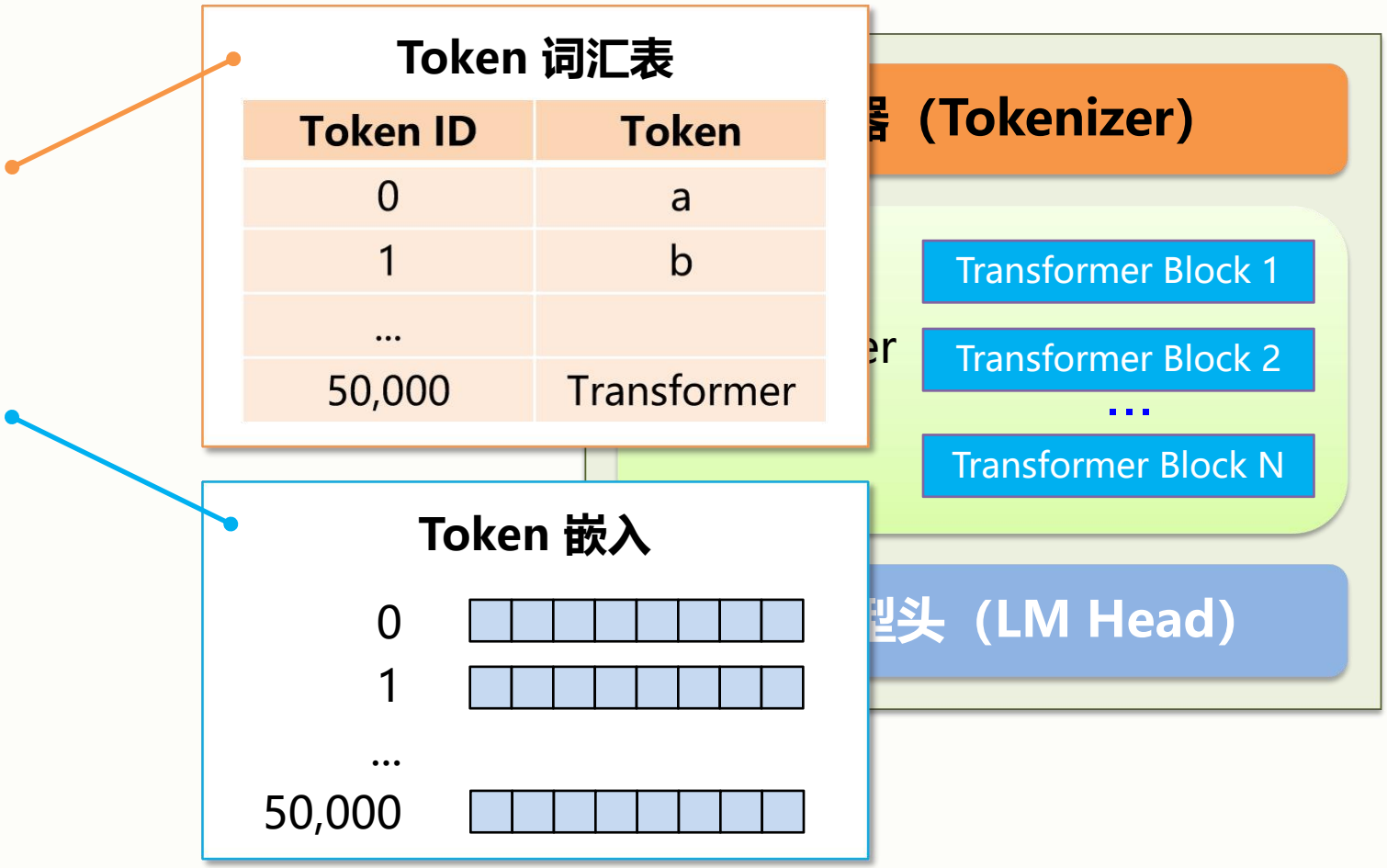
Transformer 架构的主要组件



Transformer的架构概述

Transformer 架构的主要组件

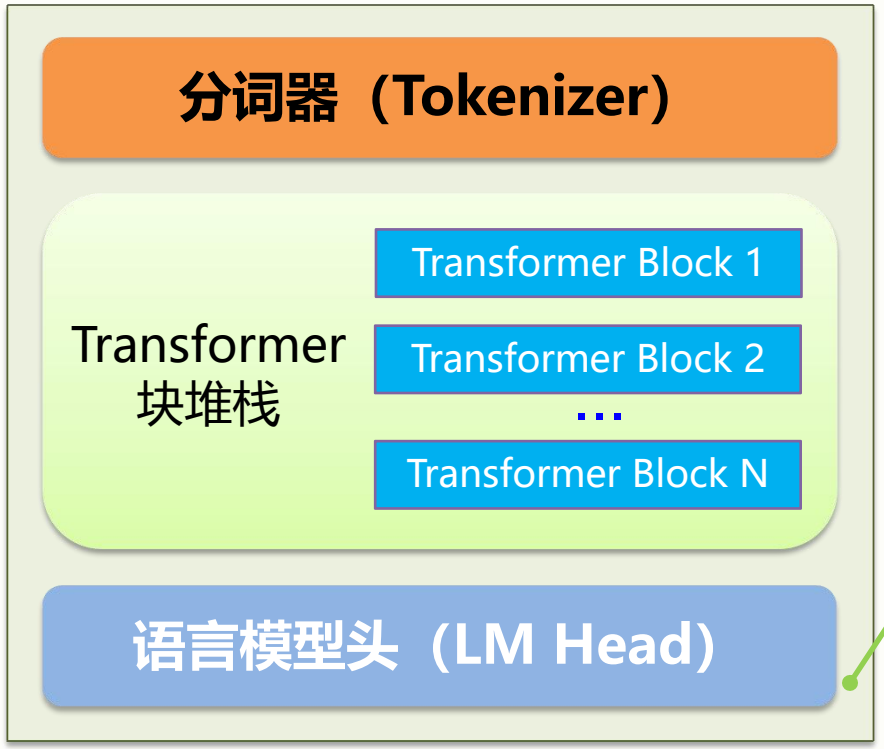
Transformer LLM



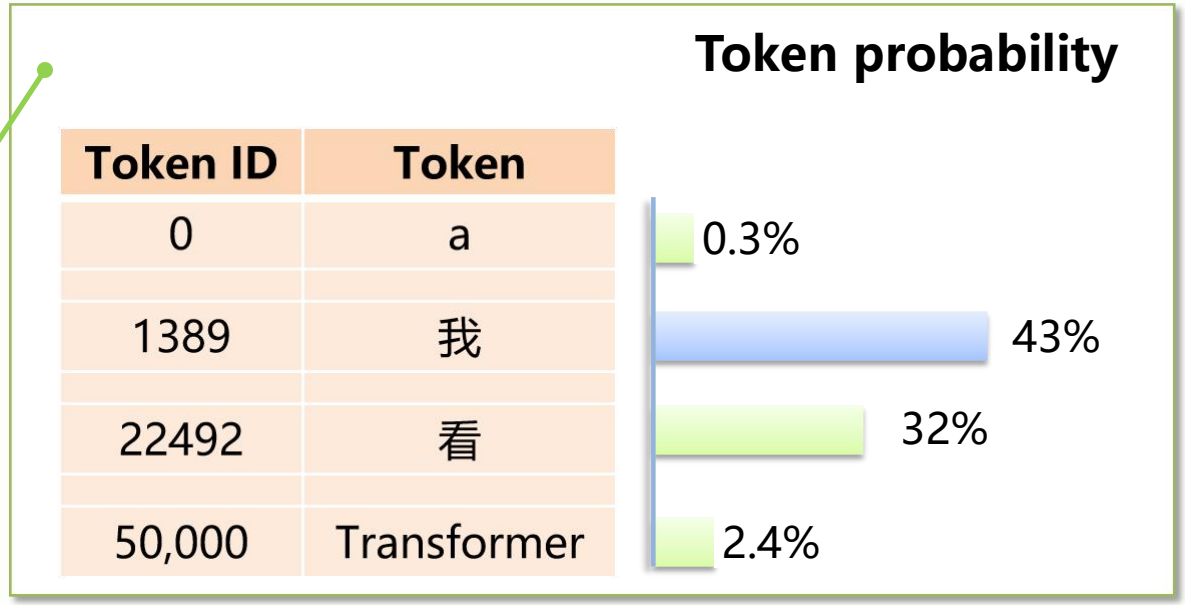
Transformer的架构概述

Transformer 架构的主要组件

Transformer LLM



$$P(y_i) = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}}$$

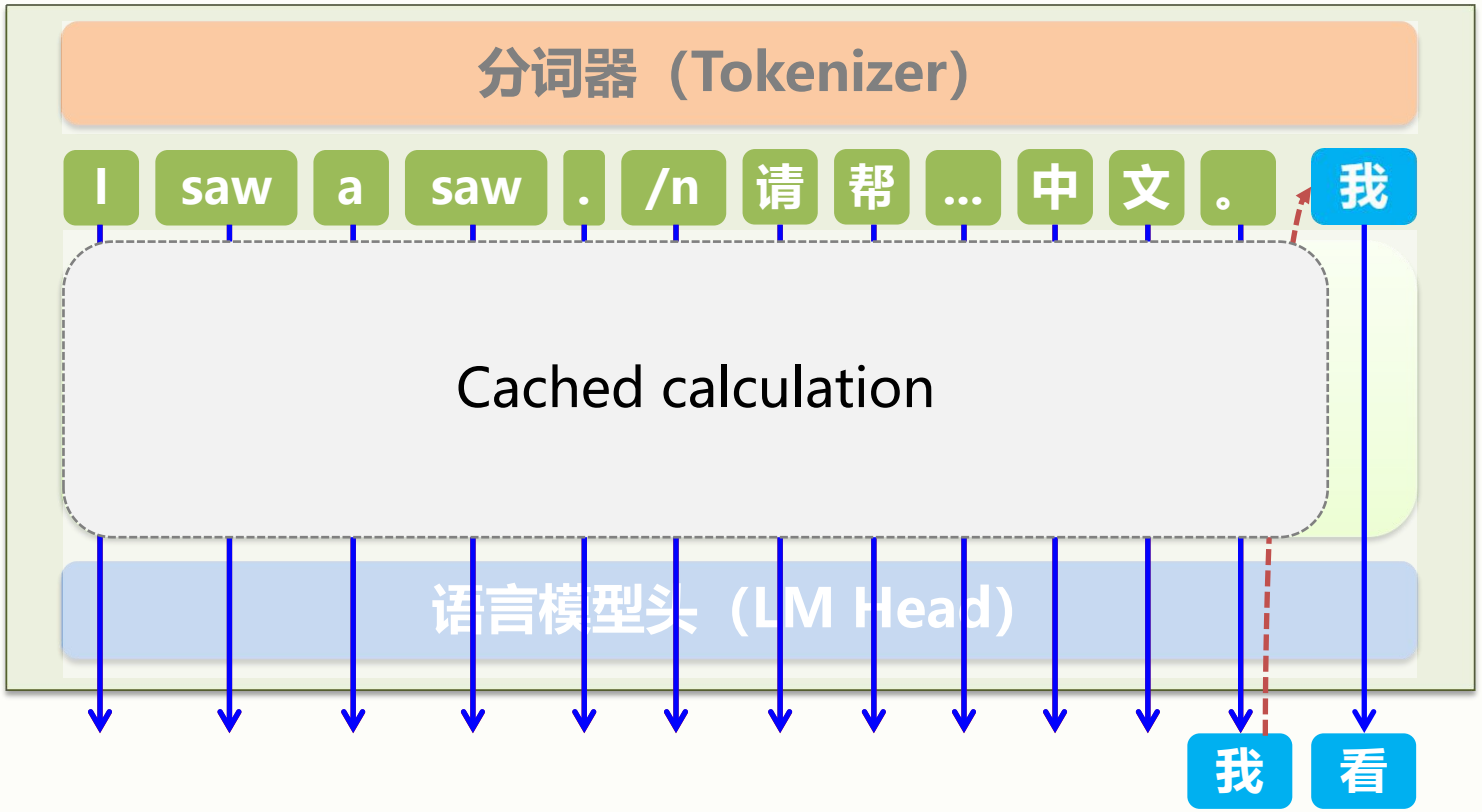


Transformer的架构概述

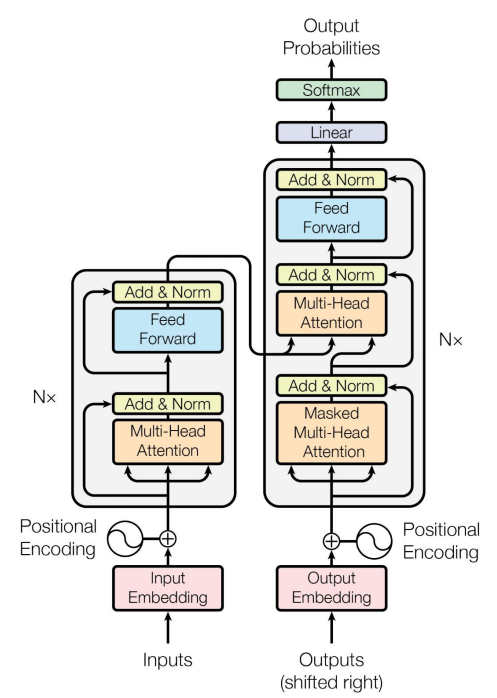
Transformer 架构的主要组件

I saw a saw.
请帮我把这个句子翻译成中文。

Transformer LLM



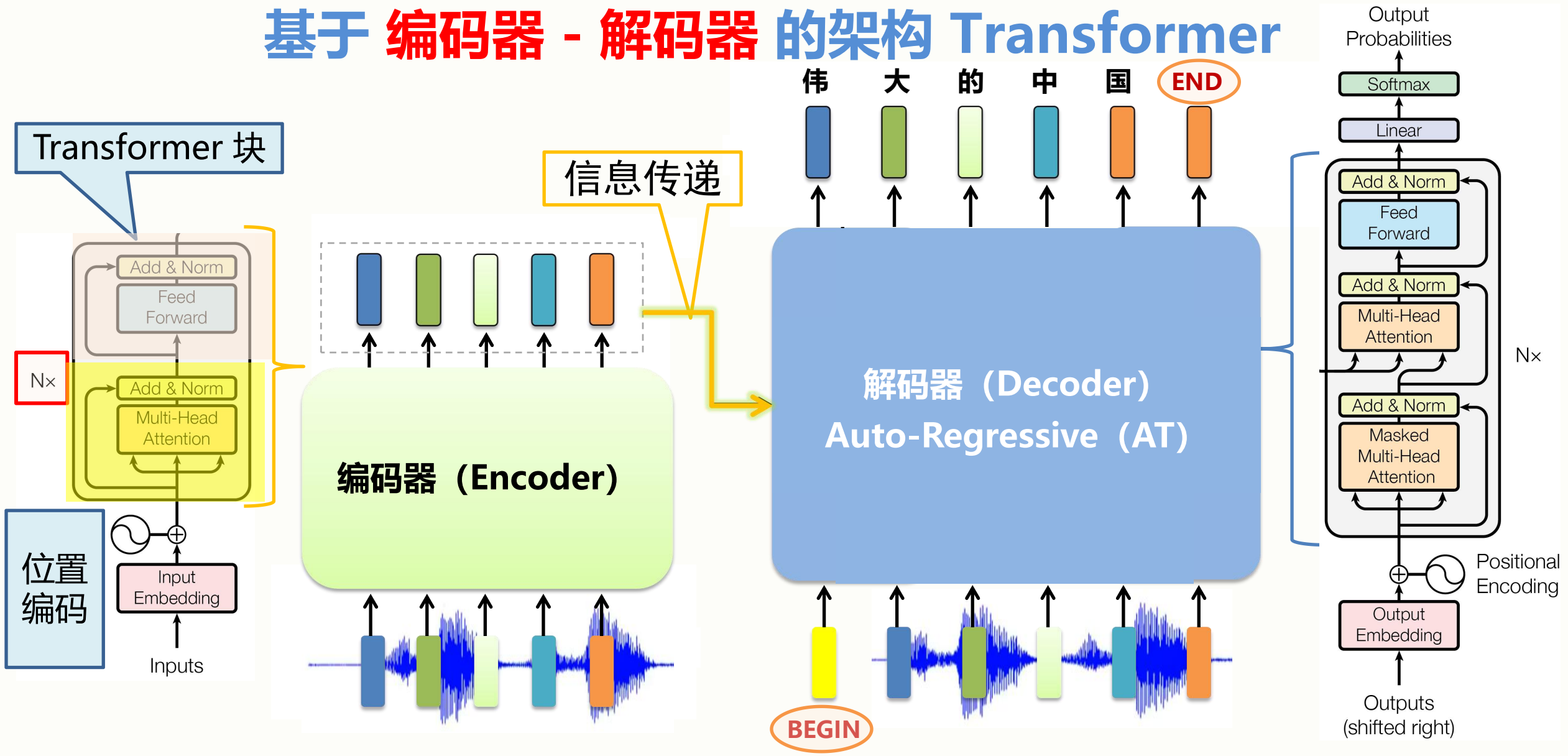
Transformer 的模块



Ahish Vaswani, Noam Shazeer, Niki Parmar, et.al. Attention is all you need. NIPS2017.

Transformer的架构概述

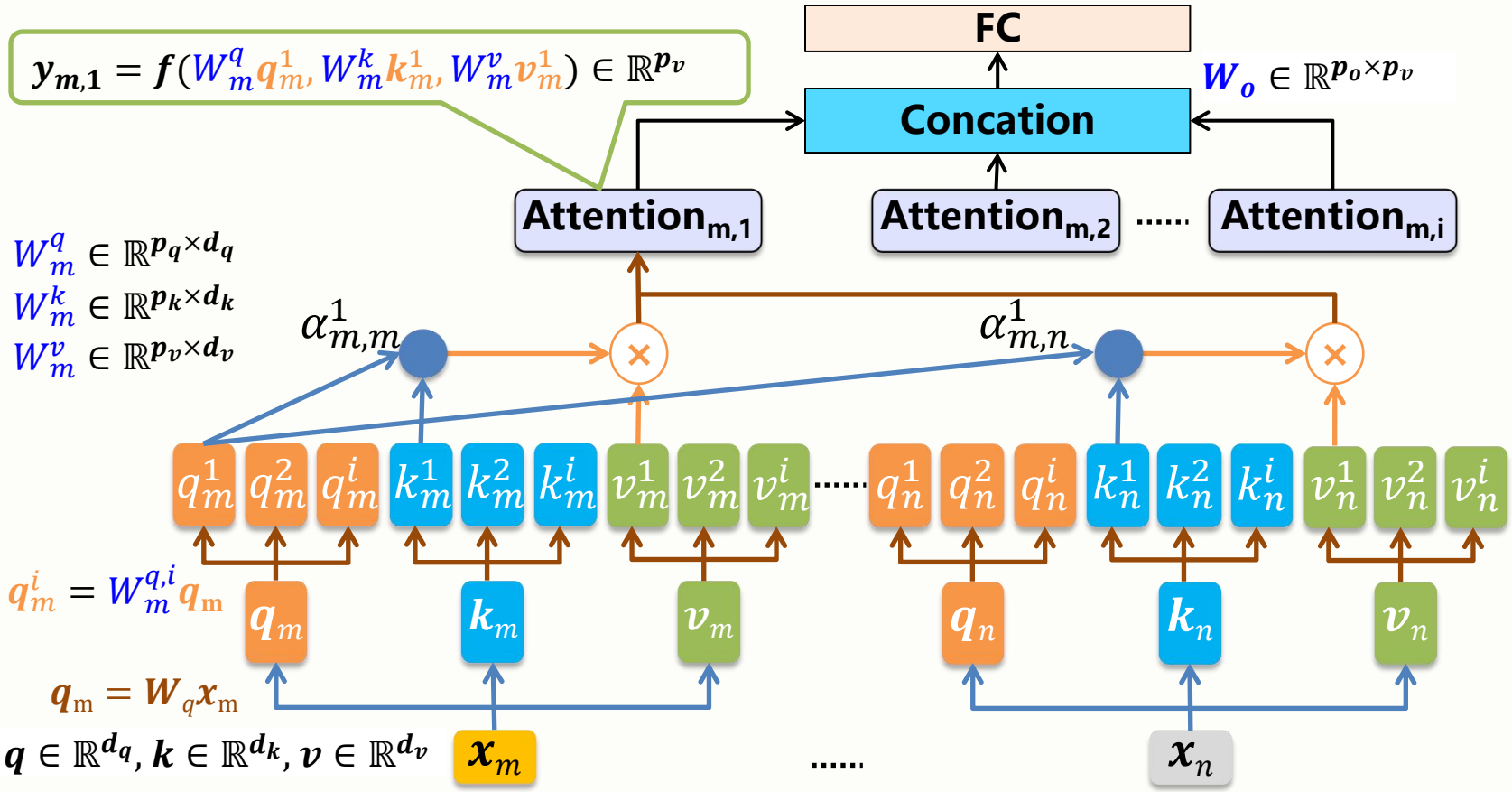
基于 编码器 - 解码器 的架构 Transformer

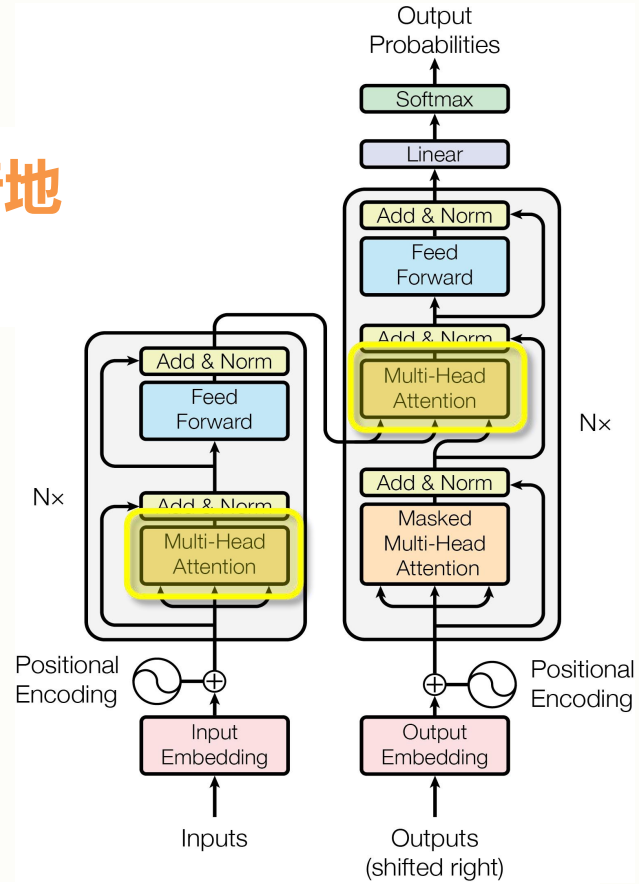


Transformer的模块

多头注意力

- **多头注意力**：对于同一组 query, key, value，通过多组线性变换并行地捕捉不同的特征。短距离关系（句子）和长距离关系（段落）。



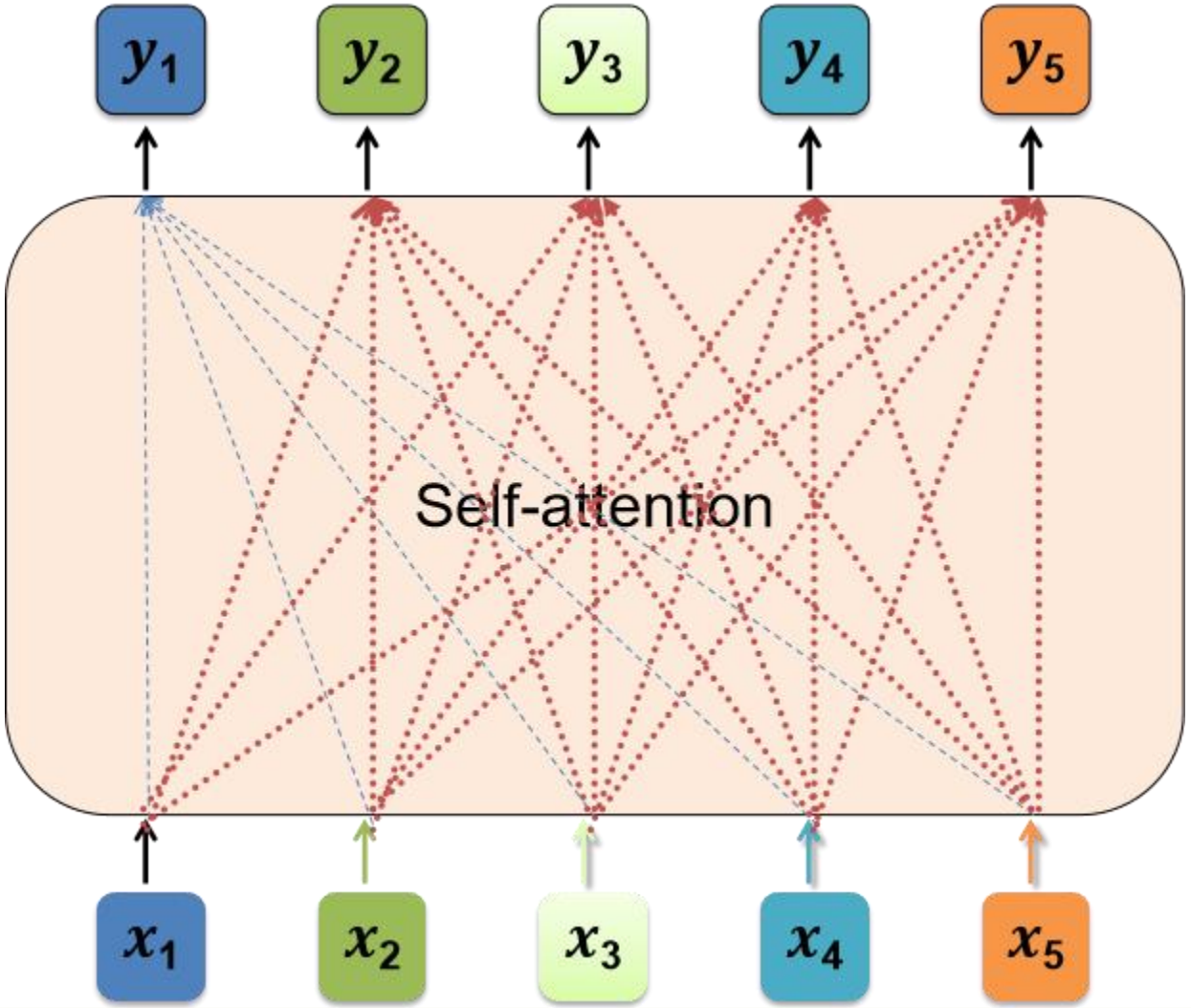


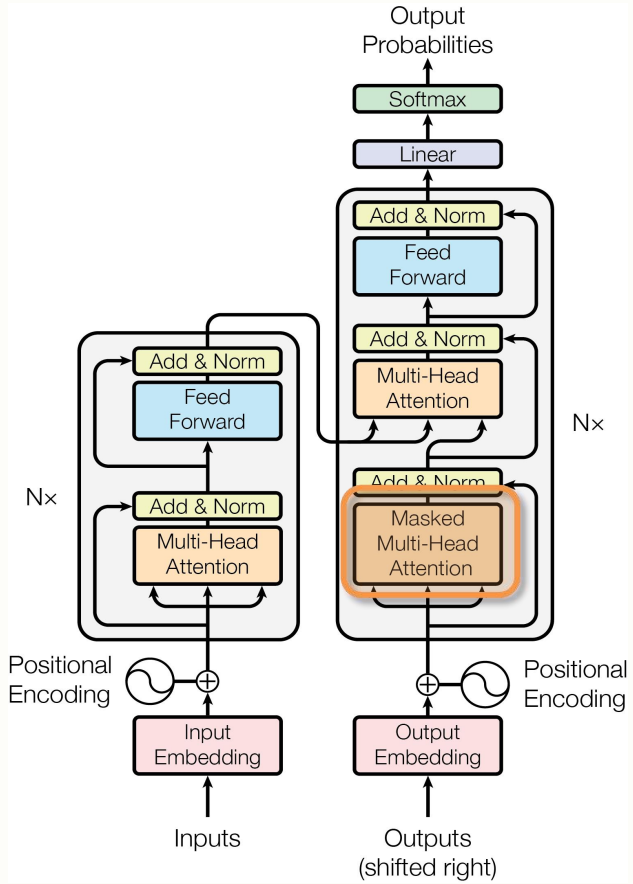
- ***多头注意力的输出***

$$y_m = W_o \begin{bmatrix} y_{m,1} \\ y_{m,2} \\ \cdots \\ y_{m,i} \end{bmatrix} \in \mathbb{R}^{p_o}$$

Transformer的模块

有掩码的多头注意力

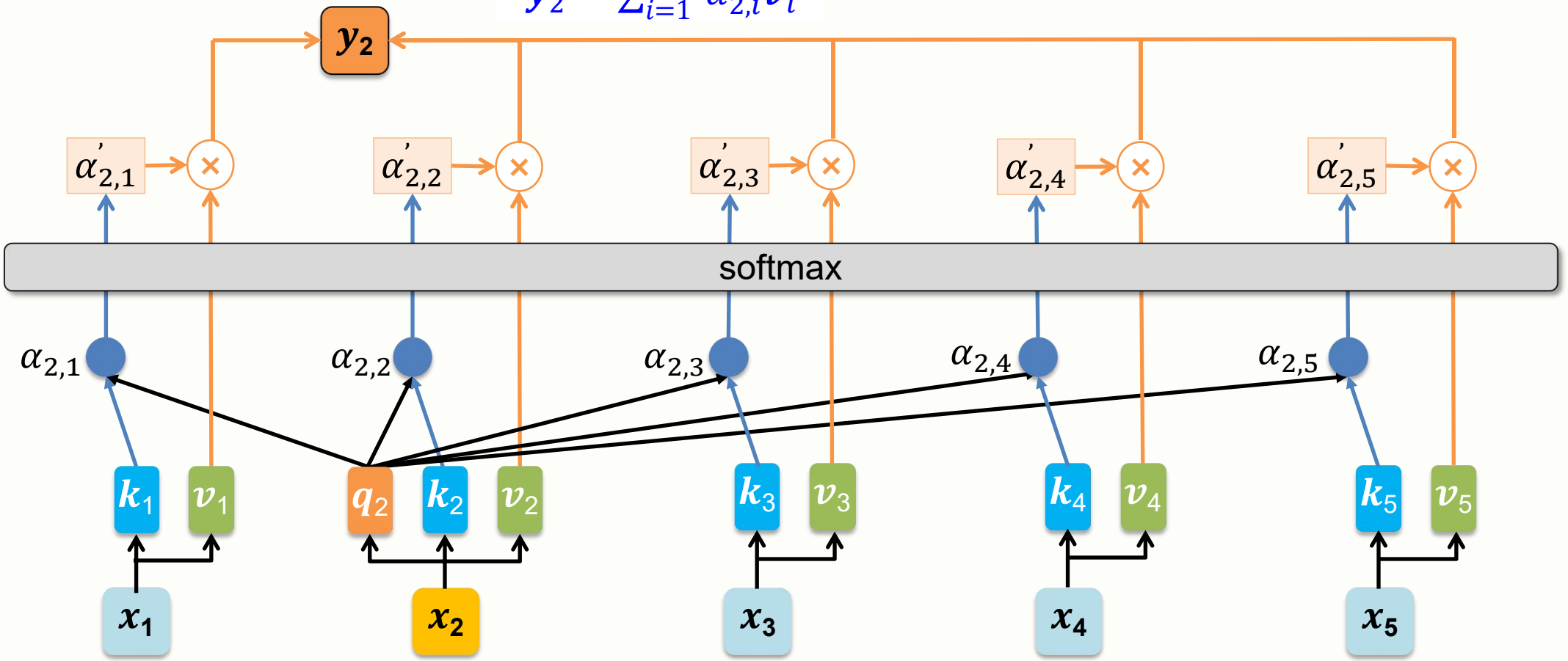




Transformer的模块

有掩码的多头注意力

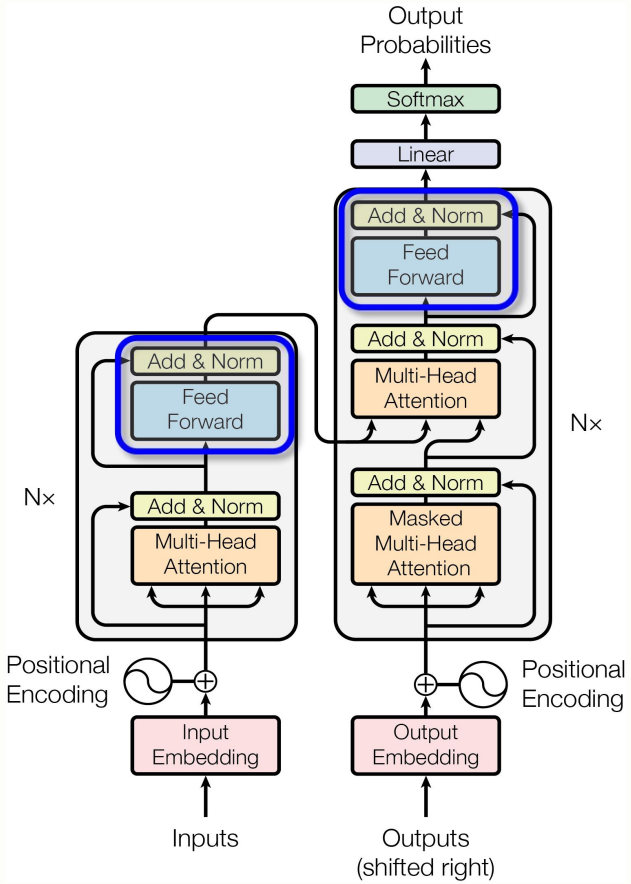
$$y_2 = \sum_{i=1}^{n=2} \alpha'_{2,i} v_i$$



Transformer的模块

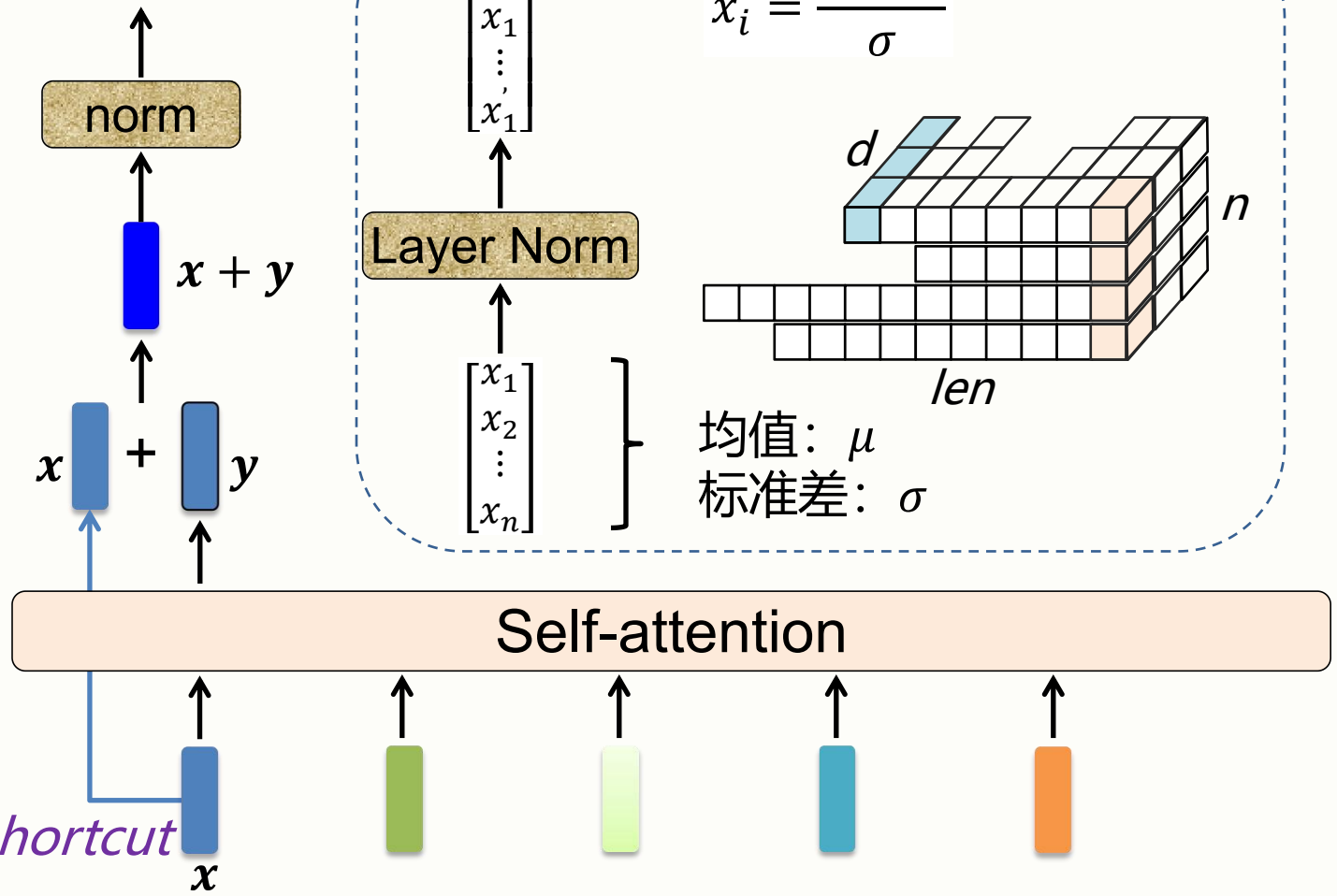
基于位置的前馈网络

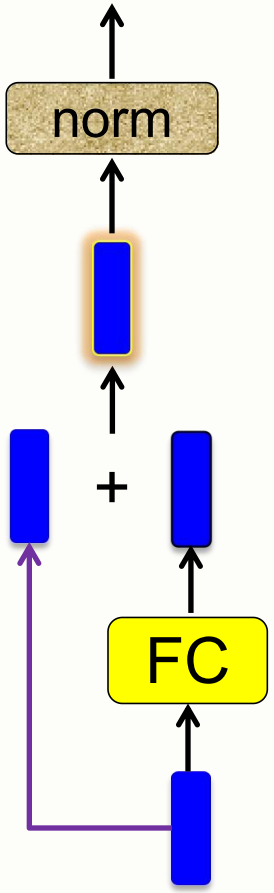
- ***卷积神经网络***
  - ✓ **输入（卷积层）形状：** （b，h，w，c）
  - ✓ **输出（全连接层）形状：** （b, h×w×c）
- ***Transformer***
  - ✓ **编码器：**将输入形状由（b，n，d）变换为（bn，d）
  - ✓ **解码器：**将输出形状由（bn，d）变回（b，n，d）
  - ✓ 等价于两个核窗口为1的一维卷积

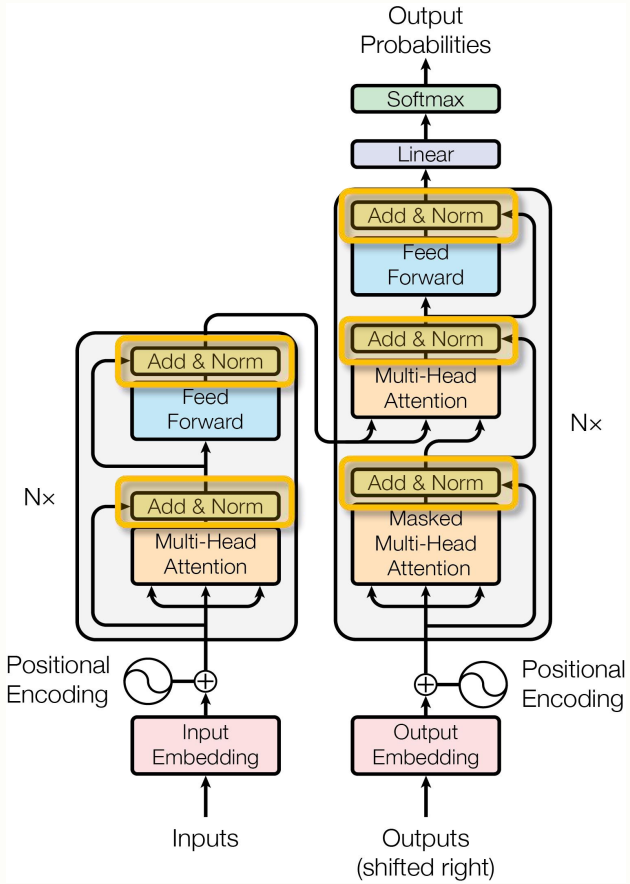


Transformer的模块

层归一化

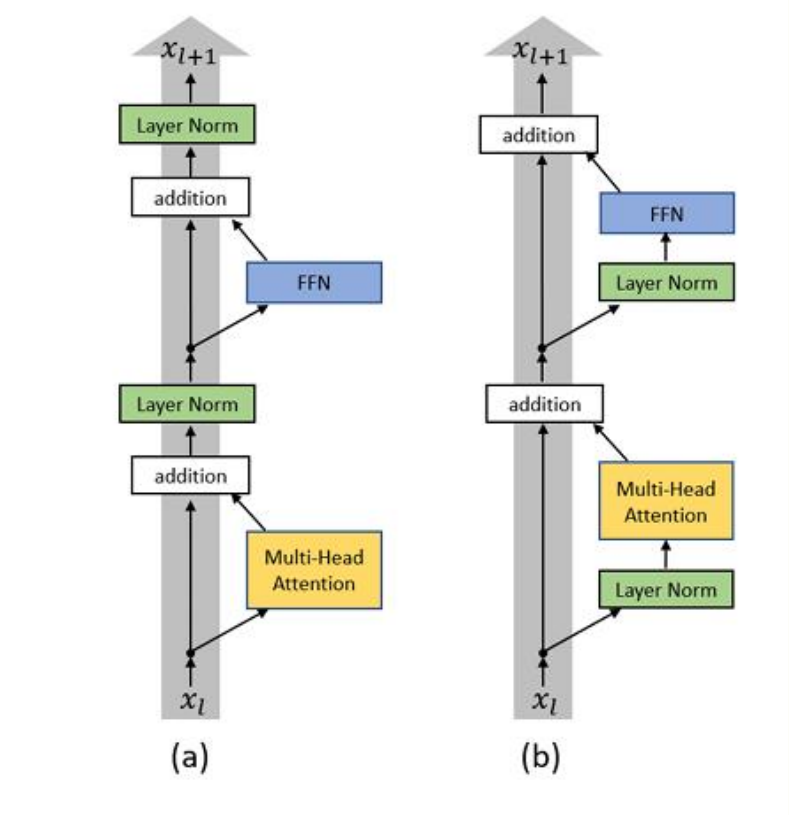


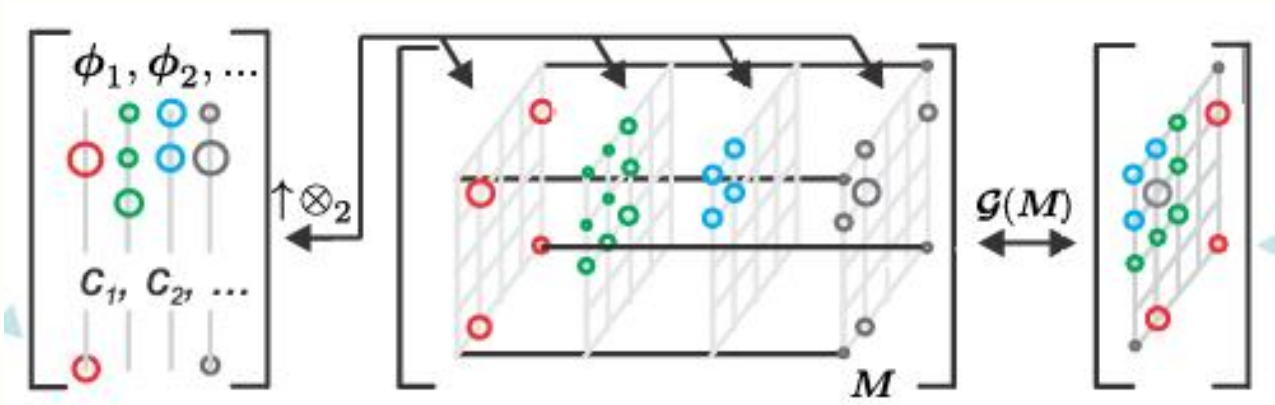




Transformer的模块

Tips1: Add & Norm

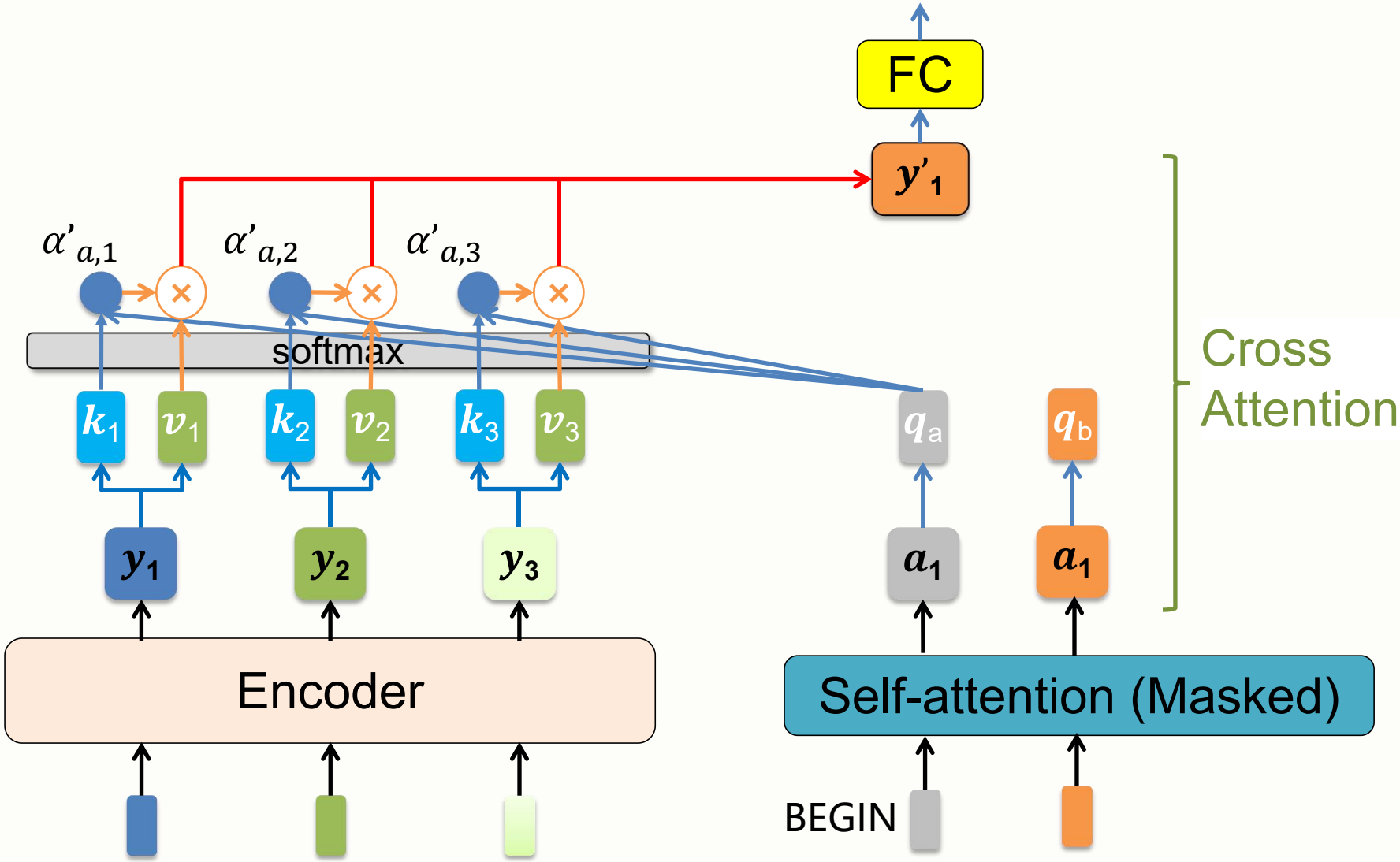


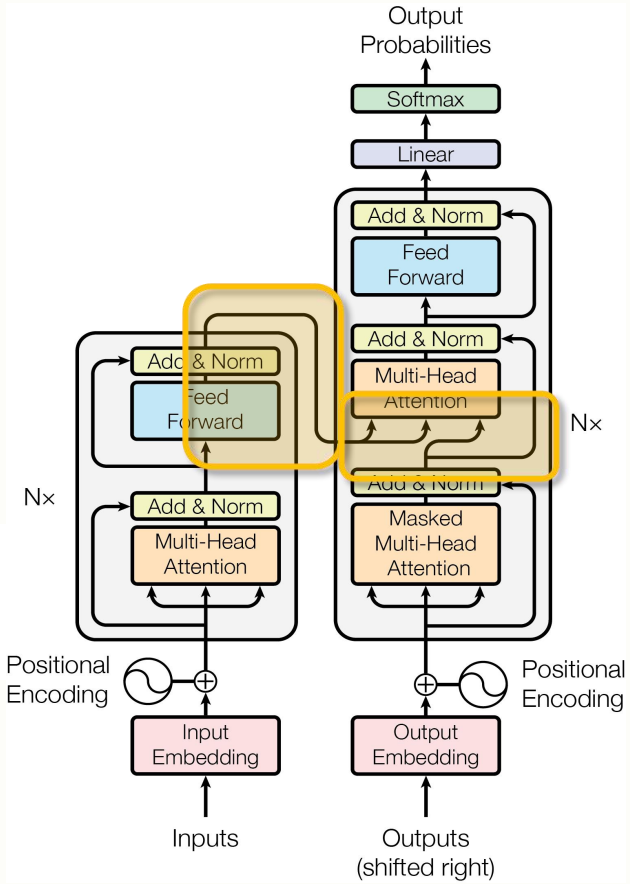


- On Layer Normalization in the Transformer Architecture
- PowerNorm: Rethinking Batch normalization in Transformers
- A Deeper Look at Power Normalizations

Transformer的模块

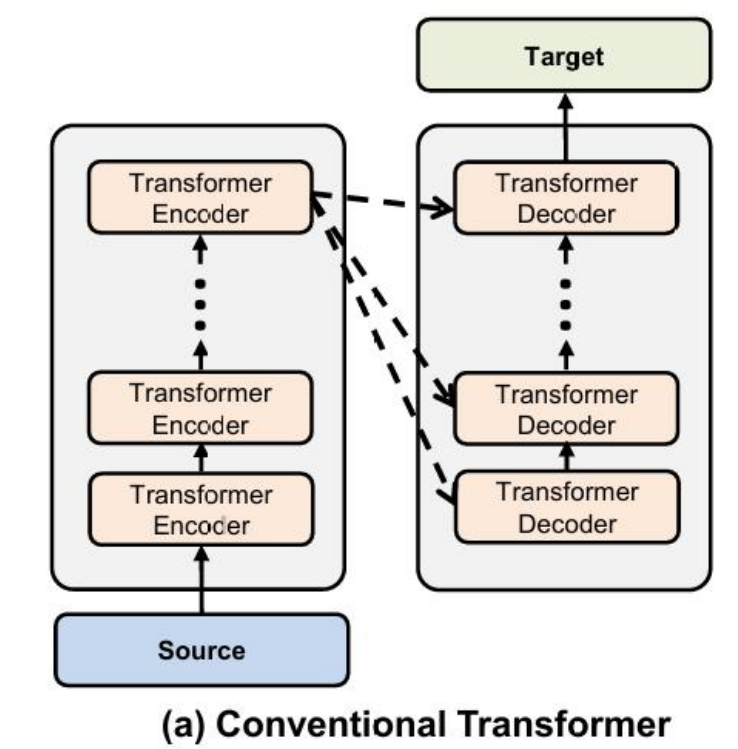
信息传递（Cross Attention）



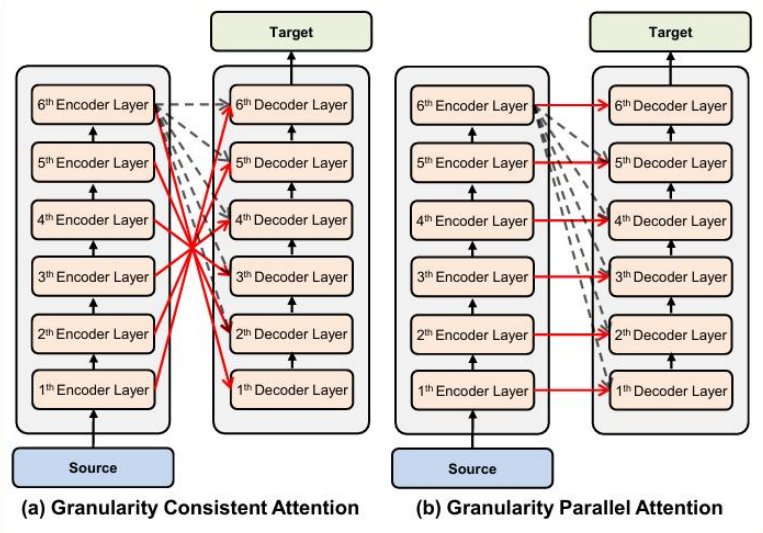


Transformer的模块

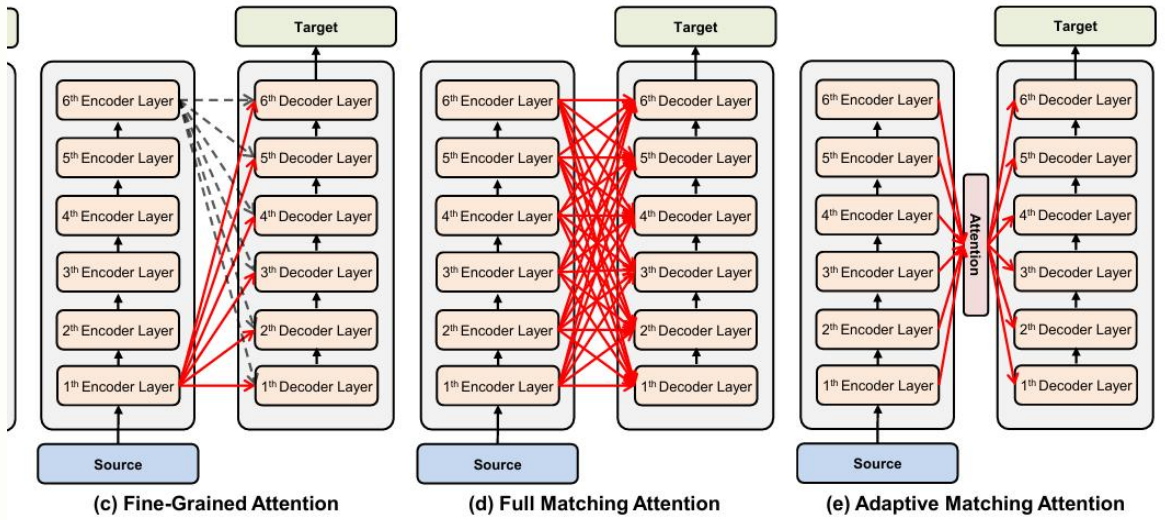
Tips2: 交叉注意力



(a) Conventional Transformer



(a) Granularity Consistent Attention (b) Granularity Parallel Attention

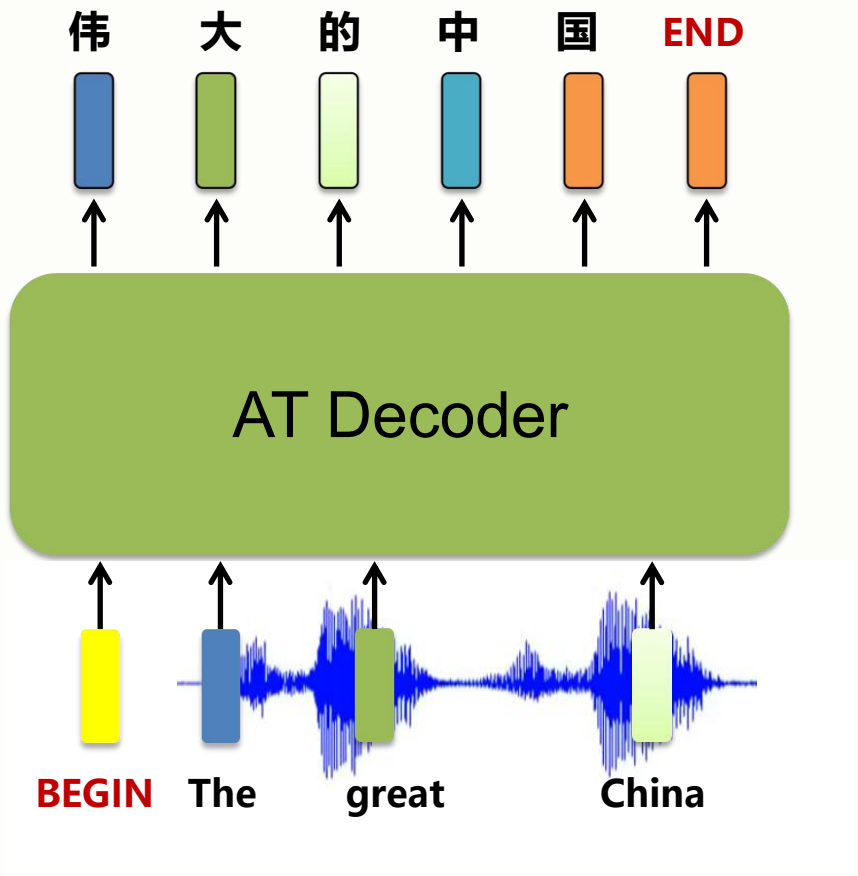


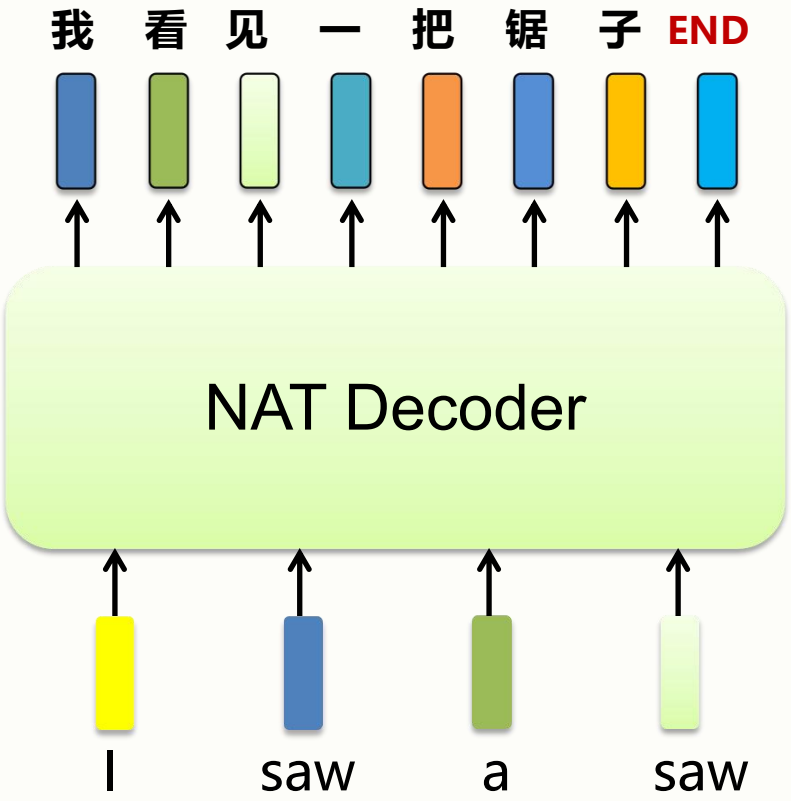
(c) Fine-Grained Attention (d) Full Matching Attention (e) Adaptive Matching Attention

- Rethinking and Improving Natural Language Generation with Layer-Wise Multi-View Decoding

Transformer的模块

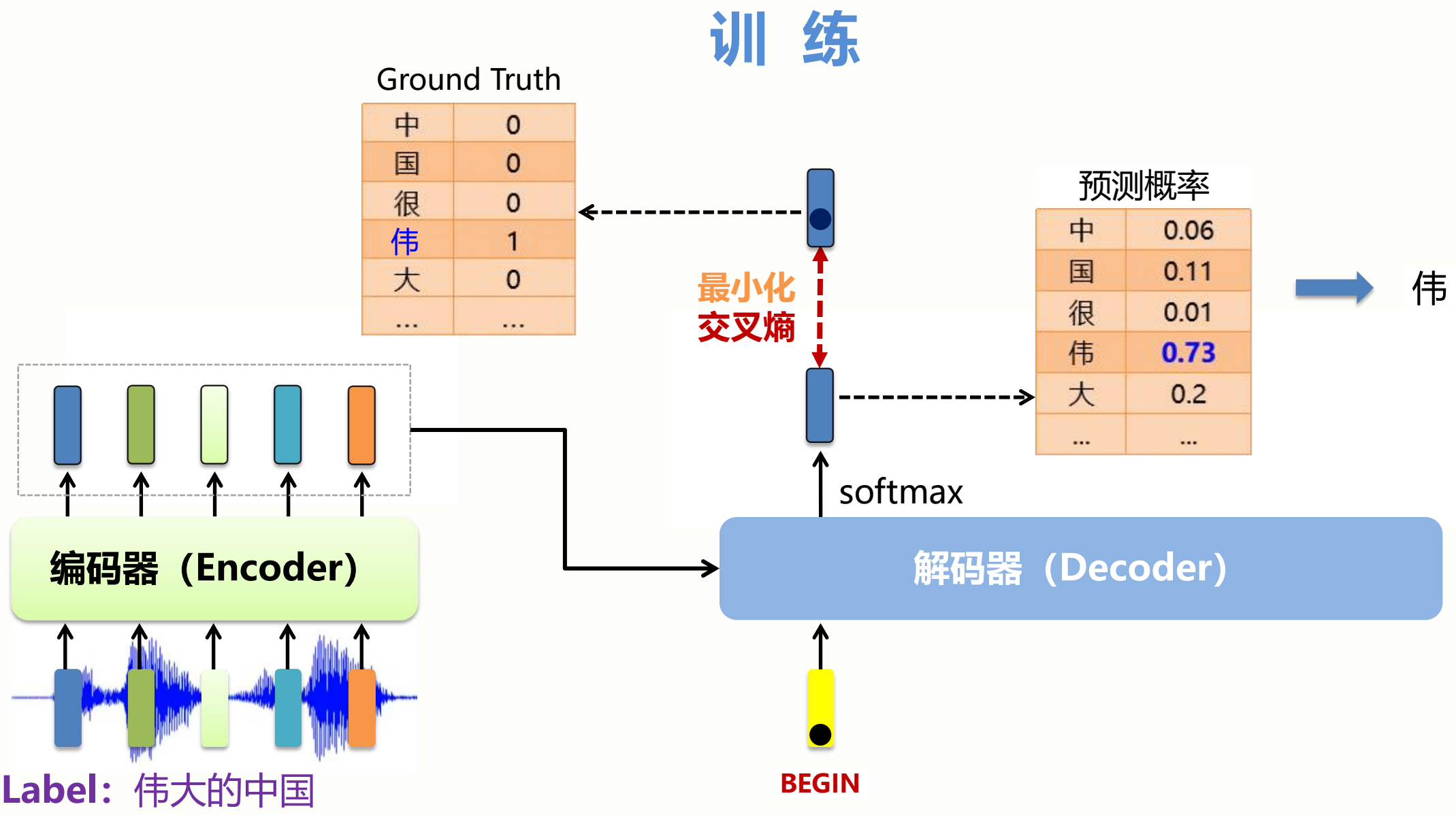
Tips3: 自回归解码器和非自回归解码器





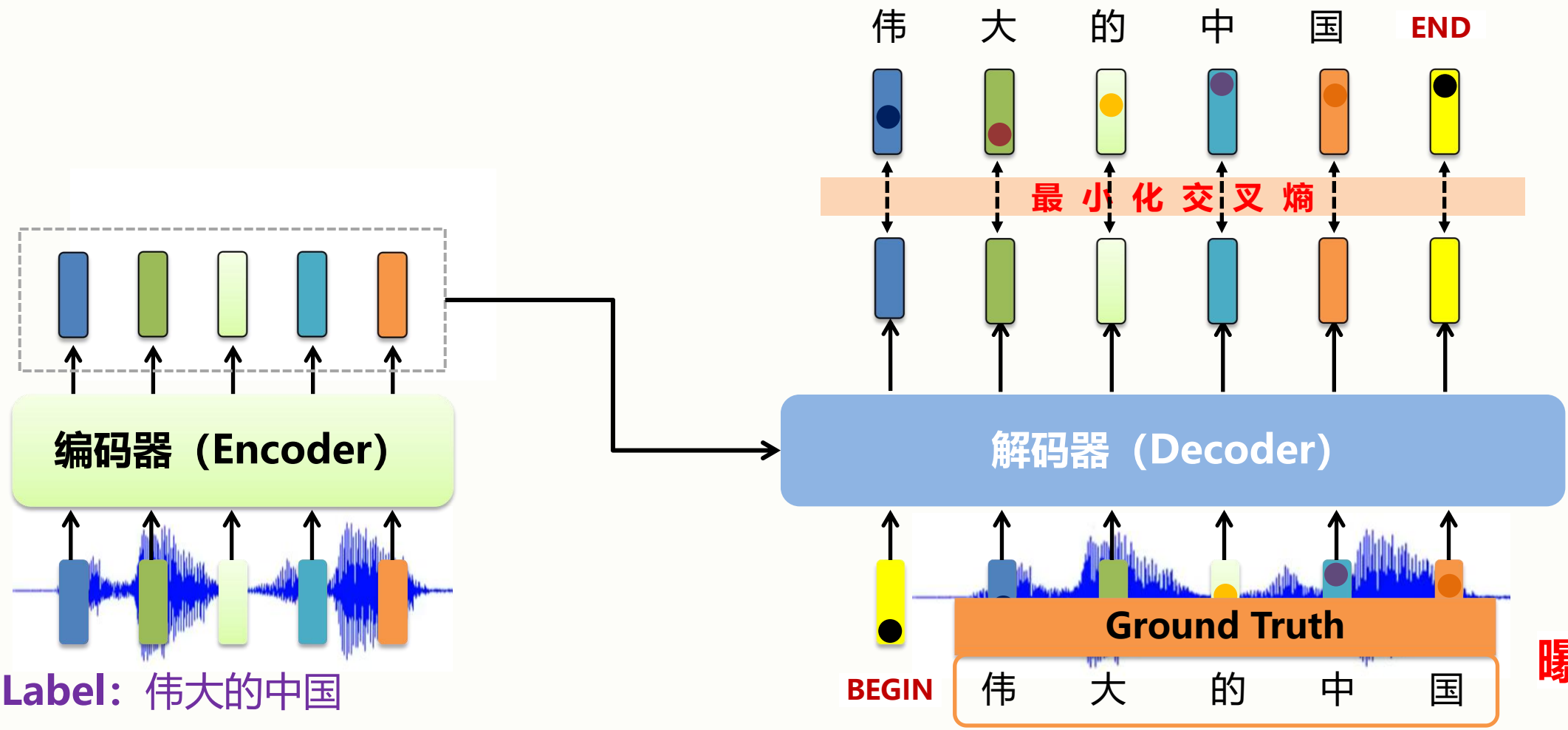
Transformer 的训练和预测

Transformer 的训练和预测



Transformer 的训练和预测

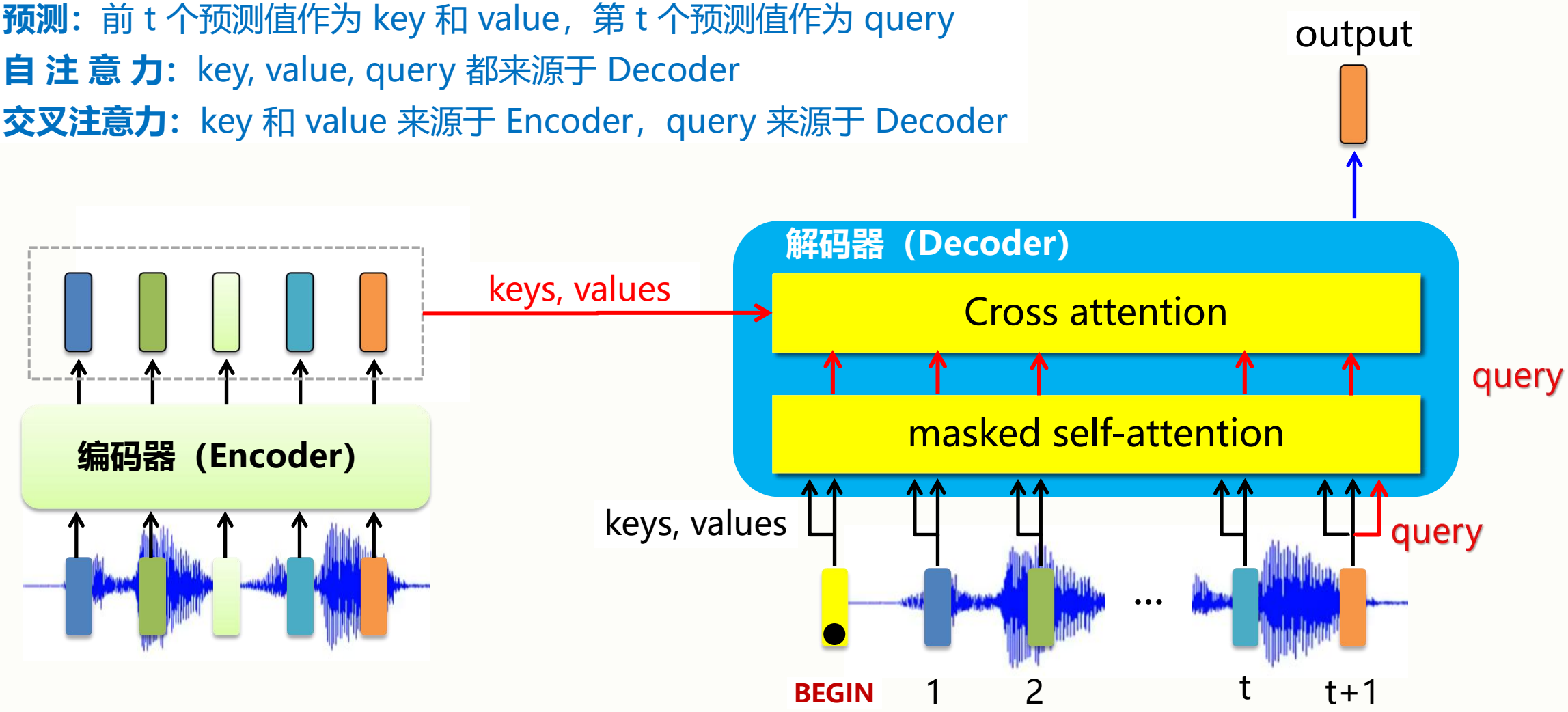
训 练



Transformer 的训练和预测

预 测

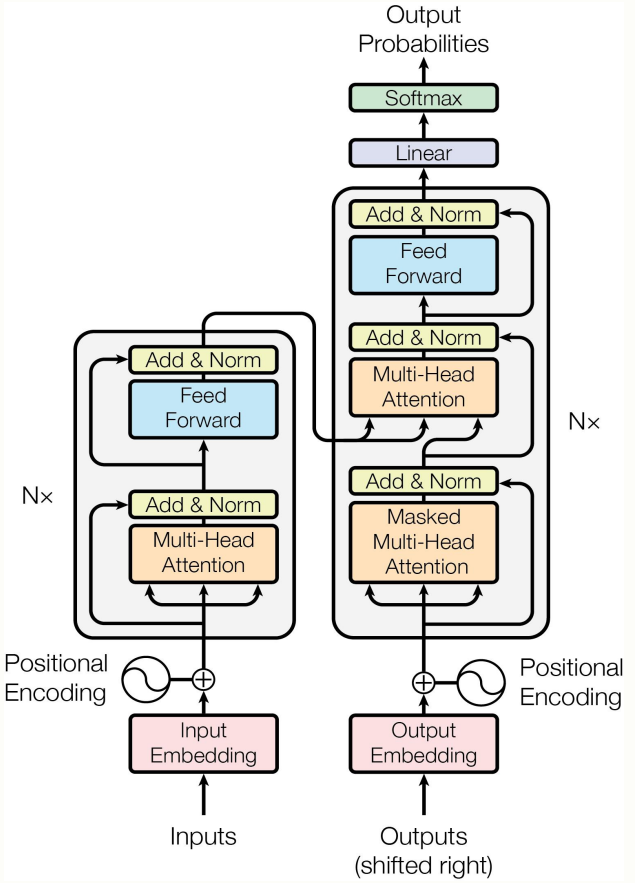
- **预测：**前 t 个预测值作为 key 和 value，第 t 个预测值作为 query
- **自 注 意 力：**key, value, query 都来源于 Decoder
- **交叉注意力：**key 和 value 来源于 Encoder，query 来源于 Decoder



Transformer

小　结

- **Transformer** 是一个纯注意力的 **编码器-解码器** 模型
- **编码器** 和 **解码器** 都由多个 transformer 块构成，主要包括 *多头自注意力*、*基于位置的前馈网络*、*层归一化组成*
- **编码器使用** self-attention 捕捉全局依赖；**解码器**采用 masked self-attention 来屏蔽未来信息，并实现历史序列信息的获取
- 编码器和解码器之间使用 cross-attention 进行连接
- Positional Encoding 通过硬植入的方式将相对位置信息融合到每一个 token 嵌入向量中。



欧老师的联系方式

读万卷书 行万里路 只为最好的修炼

欧新宇老师

QQ：14777591 (宇宙骑士)

Email：ouxinyu@alumni.hust.edu.cn

Website: http://ouxinyu.cn

Tel：18687840023

地址：安宁校区 诚远楼201

南院 智能应用研究院A306-2