Transformer for Vision

山世光

中科院计算所

sgshan@ict.ac.cn

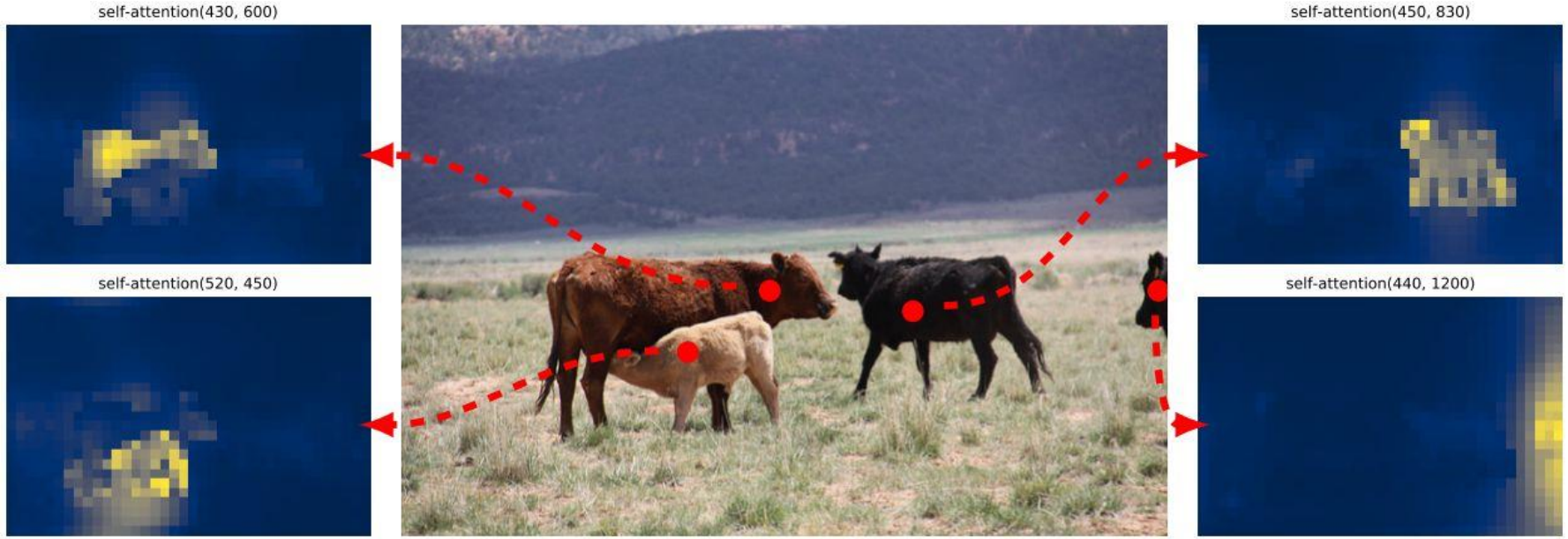
中国科学院计算技术研究所
Institute of Computing Technology,Chinese Academy of Sciences

本课件内容参考了多篇知乎文章

- https://zhuanlan.zhihu.com/p/308301901
- https://zhuanlan.zhihu.com/p/82312421
- https://zhuanlan.zhihu.com/p/336352895
- https://zhuanlan.zhihu.com/p/266069794

概述

- Transformer是一个Sequence to Sequence model
  - 特别之处在于它大量用到self-attention
  - 相比CNN，它便捷实现长程依赖关系学习
  - 相比RNN/LSTM，它可以并行处理



Sequence to Sequence模型

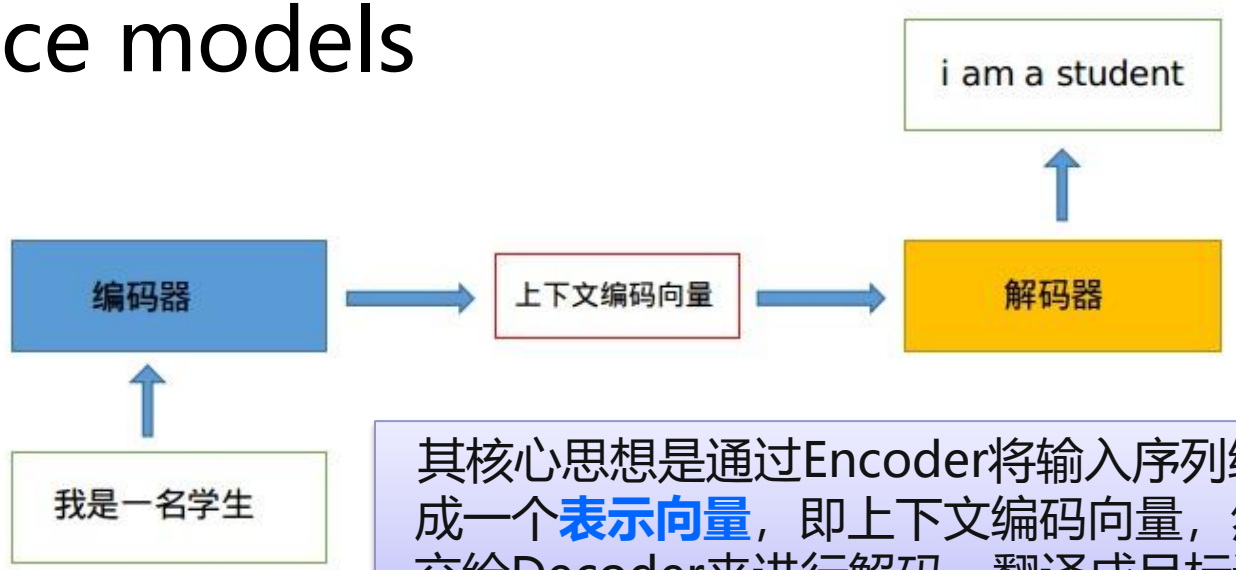
- 问题描述

本课件主要内容来自知乎文章：Vision Transformer 超详细解读 (原理分析+代码解读)，作者：科技猛兽

Sequence to Sequence模型

- 问题描述

- Sequence to sequence models
  - RNN
  - LSTM
  - GRU

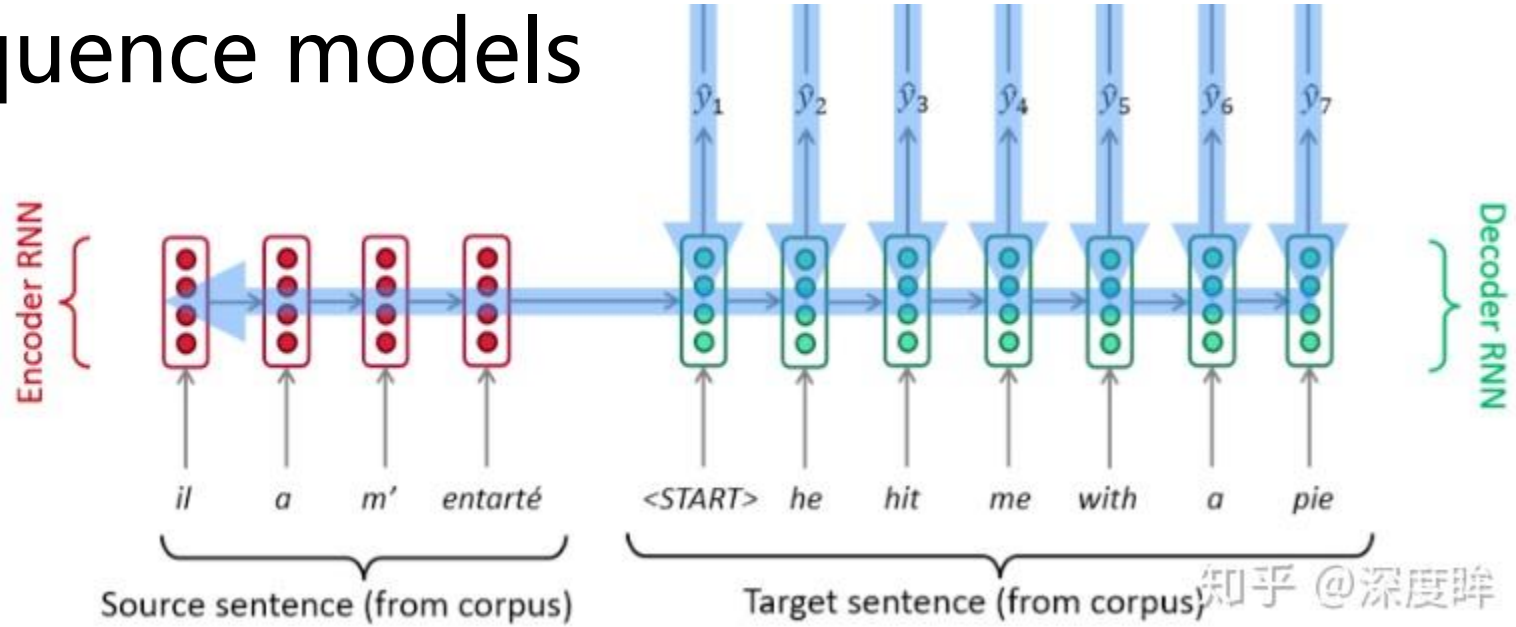


其核心思想是通过Encoder将输入序列编码成一个**表示向量**，即上下文编码向量，然后交给Decoder来进行解码，翻译成目标语言

Sequence to Sequence模型

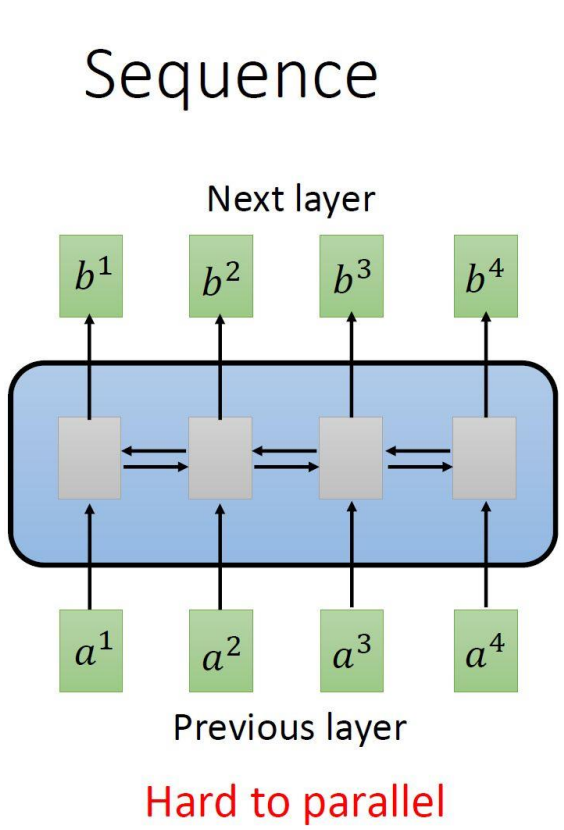
- 问题描述

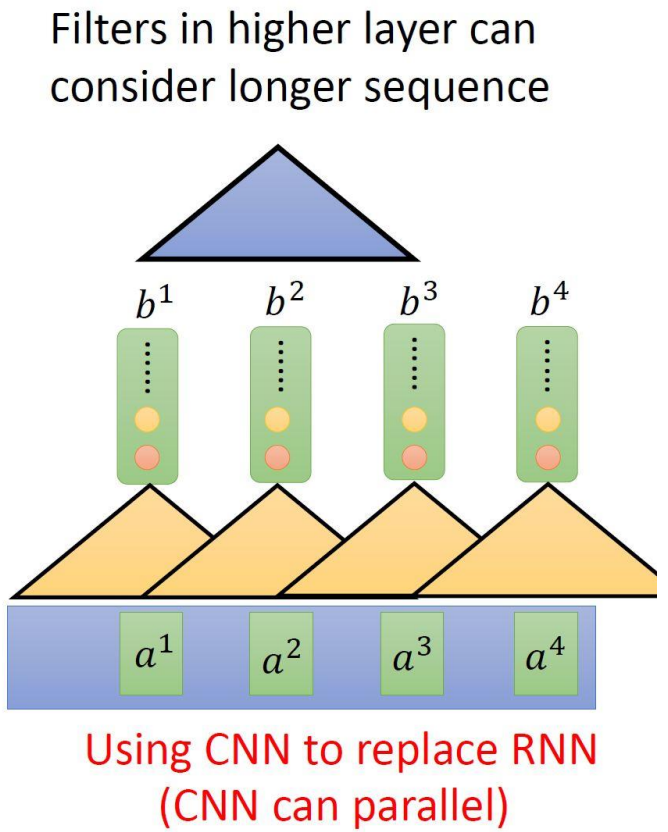
- Sequence to sequence models
  - RNN
  - LSTM
  - GRU



Sequence to Sequence模型

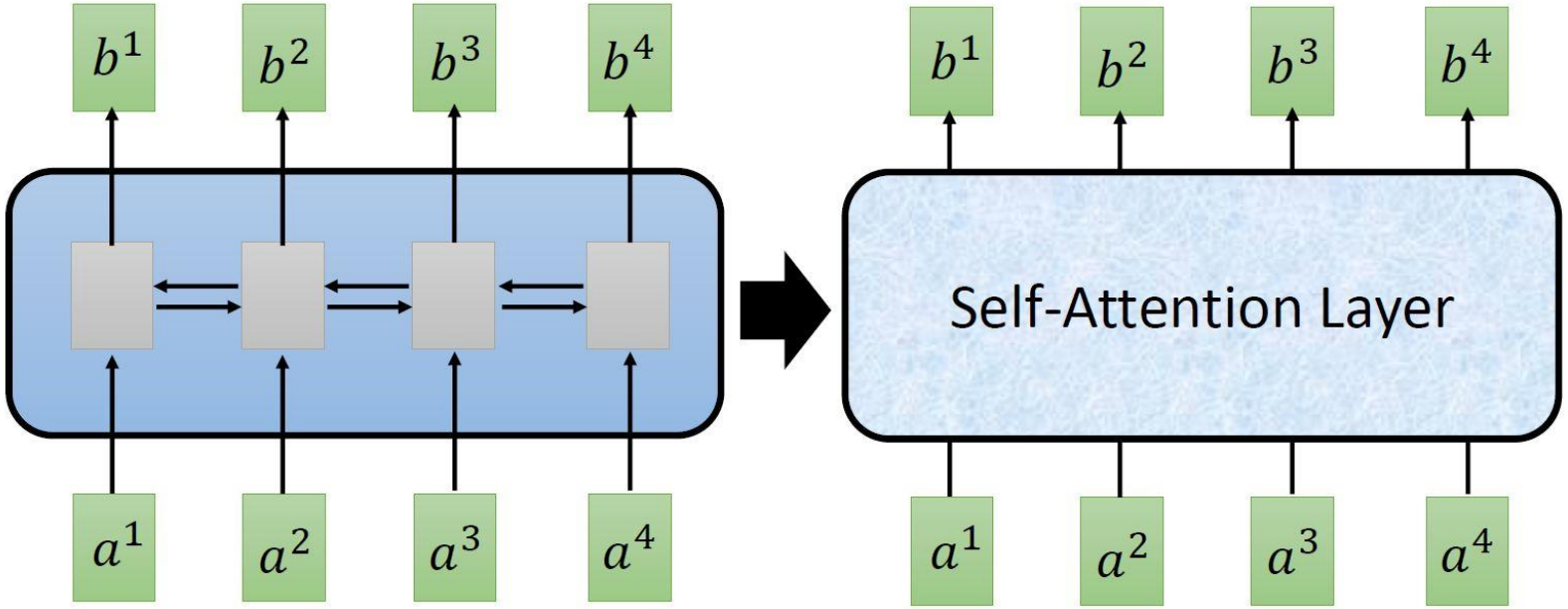
- RNN等：**难以并行**
  - 单向
    - B1输出时没见过后续输入
  - 双向
- 用CNN取代RNN
  - 层叠可以实现长程依赖





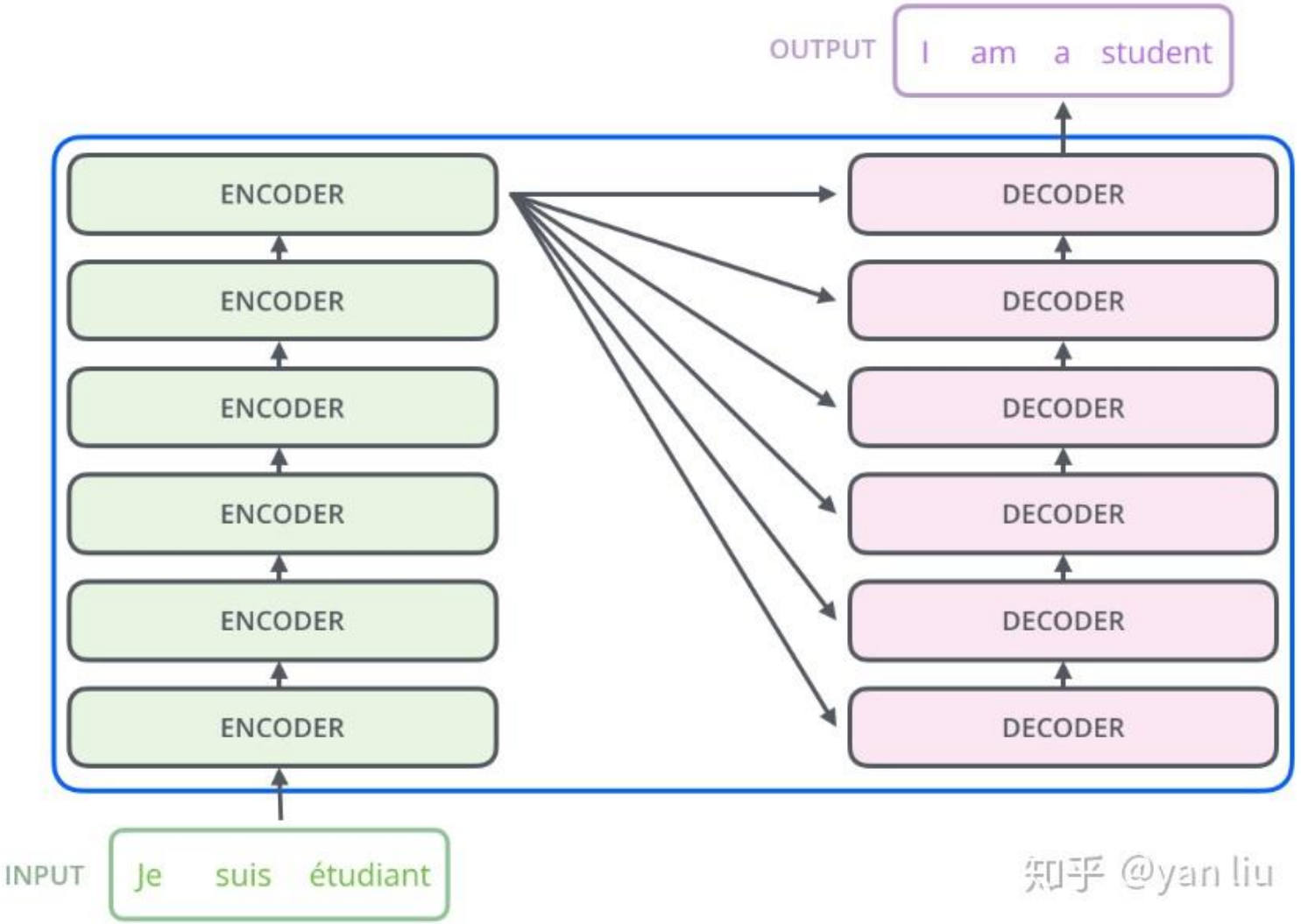
Self-Attention

- 特点：**每个输出都看过了整个输入sequence**
  - 与bi-directional RNN相同
  - 它的每一个输出可以并行化计算



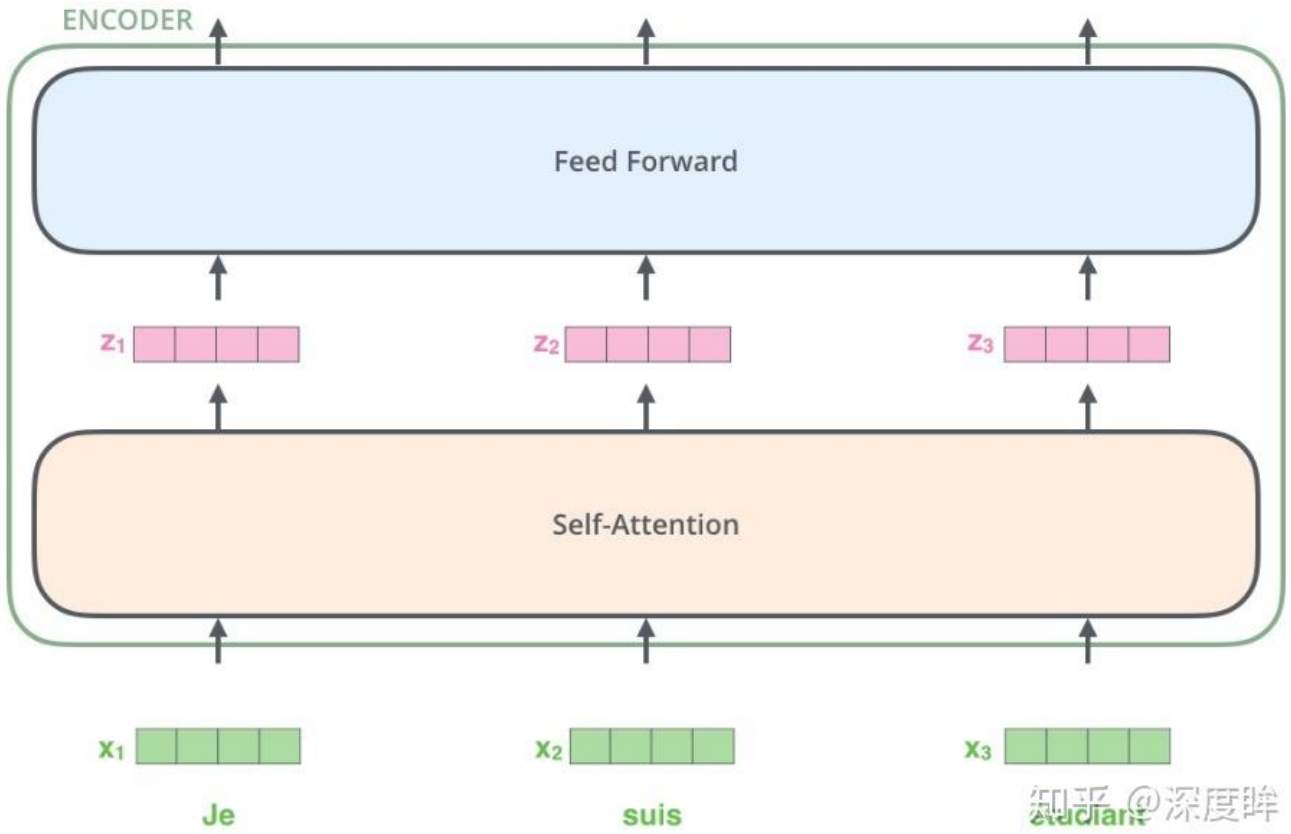
Sequence to Sequence模型

- Transformer架构



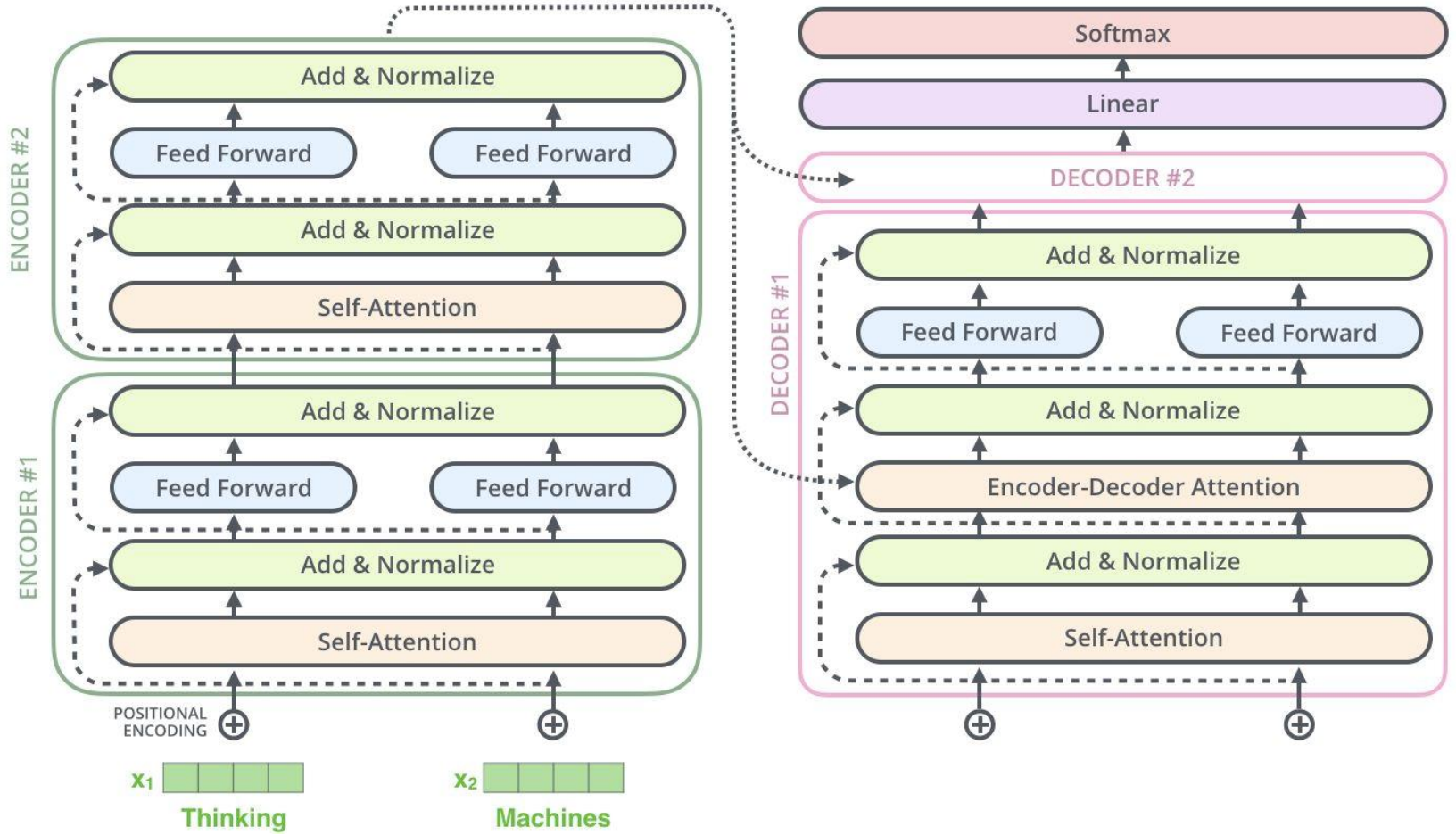
Sequence to Sequence模型

- Transformer架构
  - Encoder架构



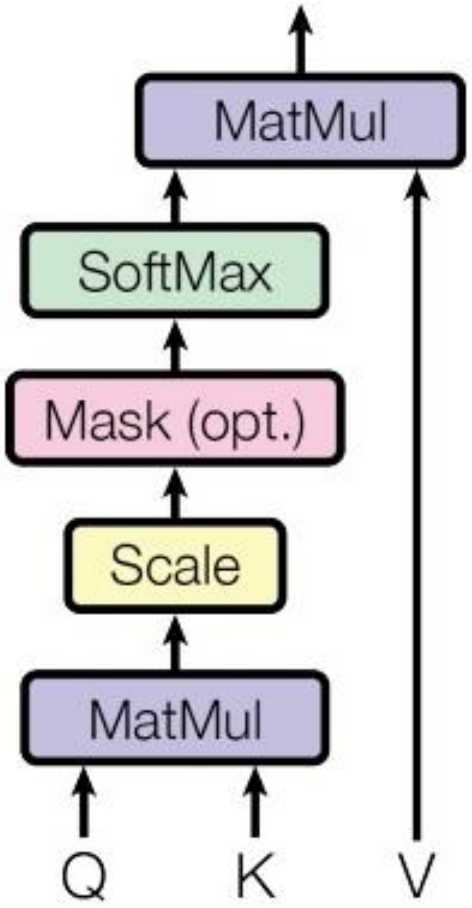
Sequence to Sequence模型

- Transformer架构
  - More details



Self-Attention

- Scaled Dot-product Attention



q: query (to match others)

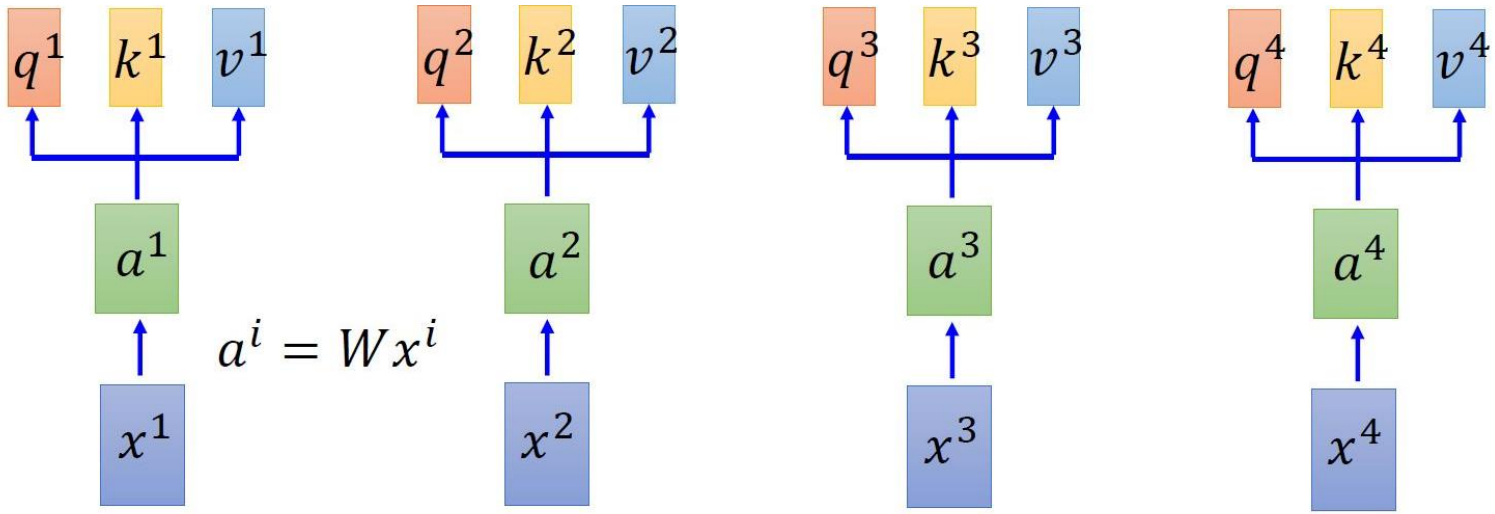
$$q^i = W^q a^i$$

k: key (to be matched)

$$k^i = W^k a^i$$

v: information to be extracted

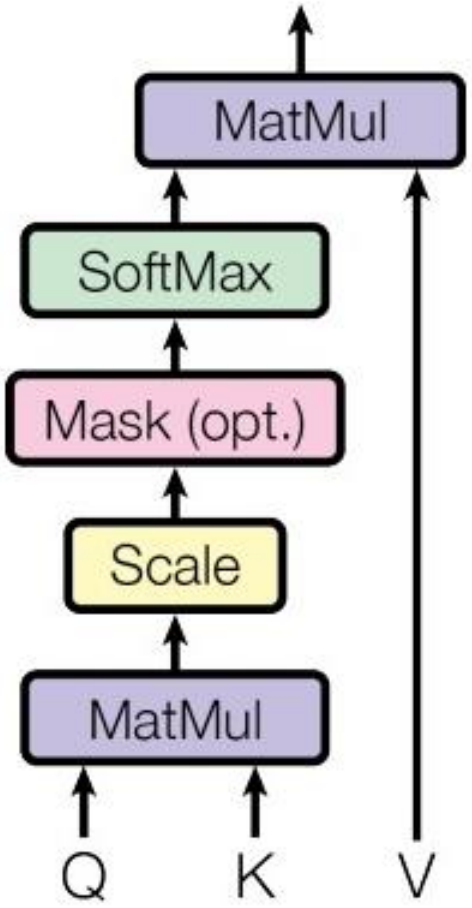
$$v^i = W^v a^i$$

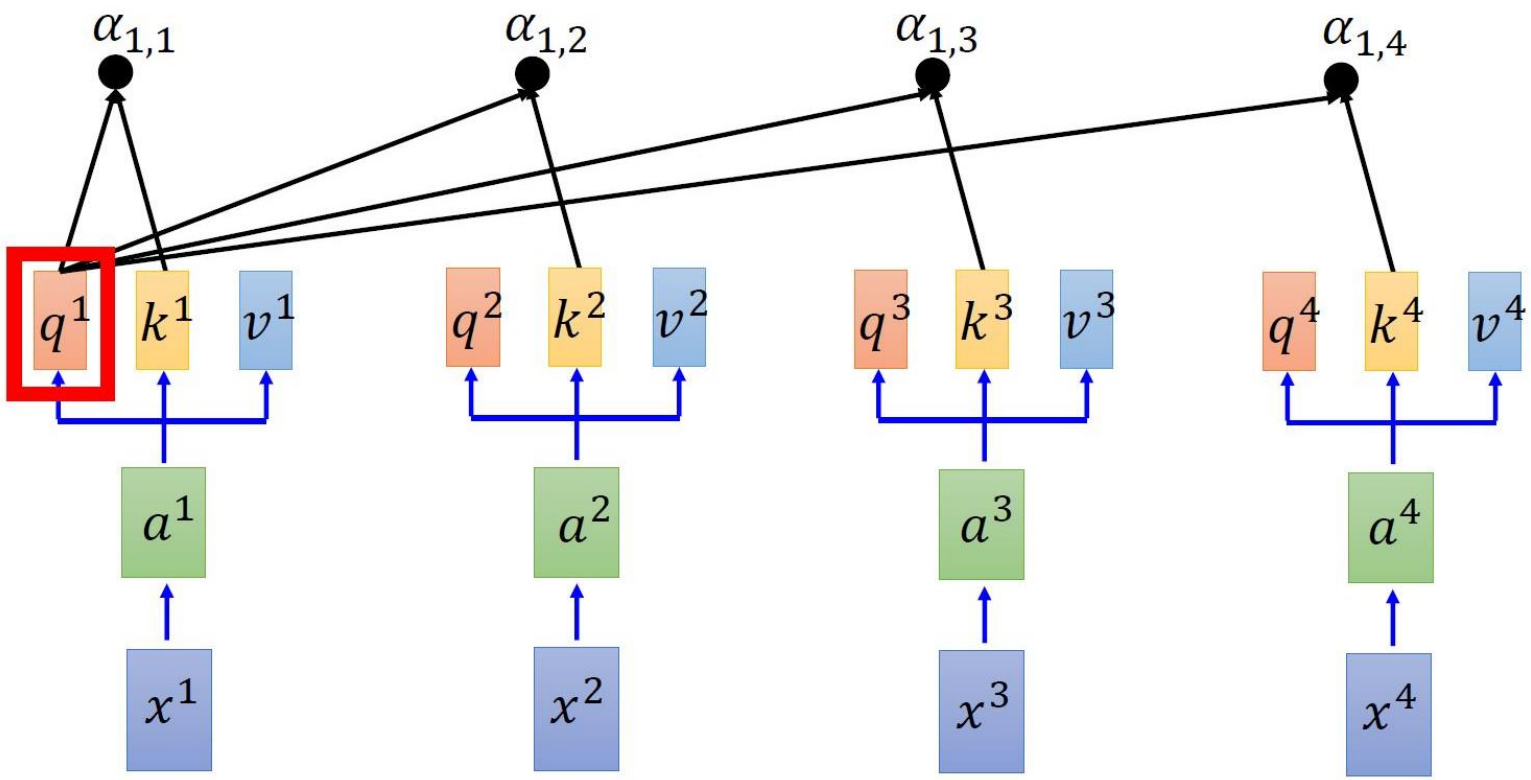


Self-Attention

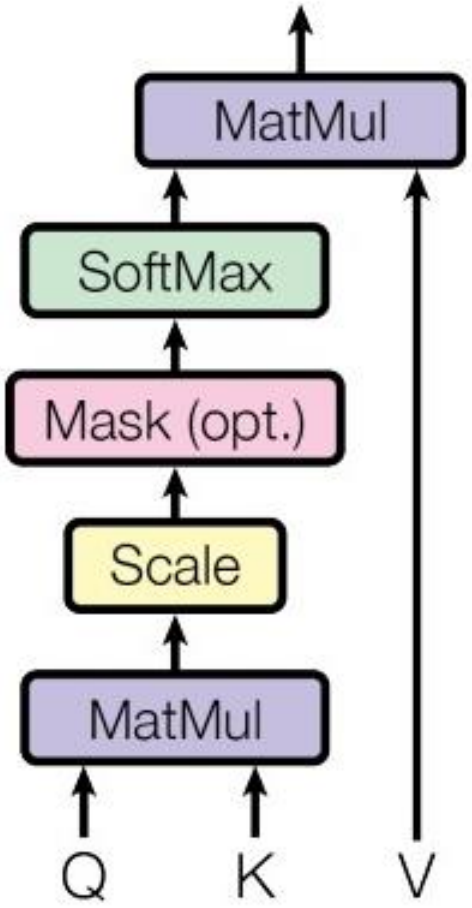
- Scaled Dot-product Attention

$$\alpha_{1,i} = \underbrace{q^1 \cdot k^i}_{\text{dot product}} / \sqrt{d}$$



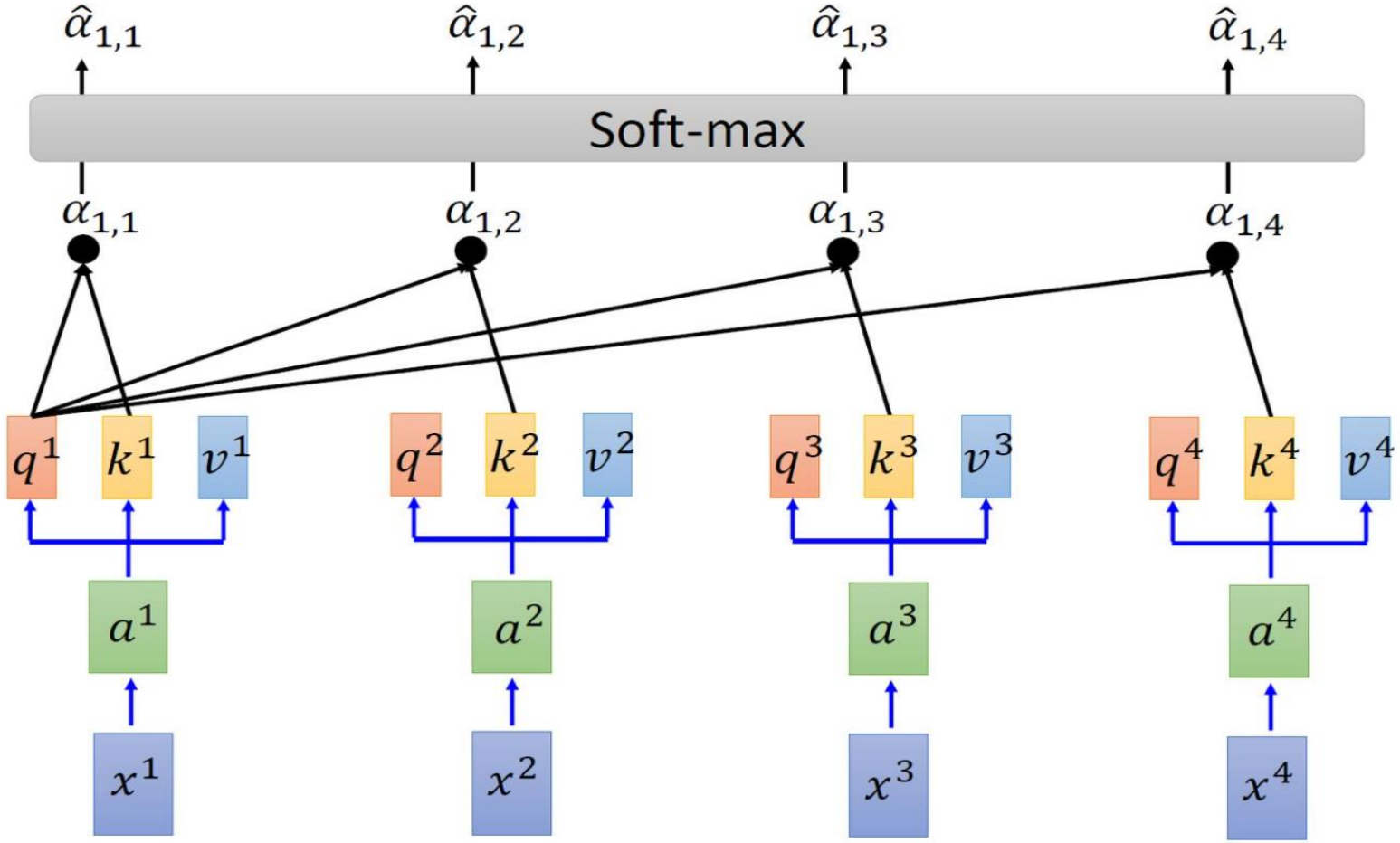


Self-Attention



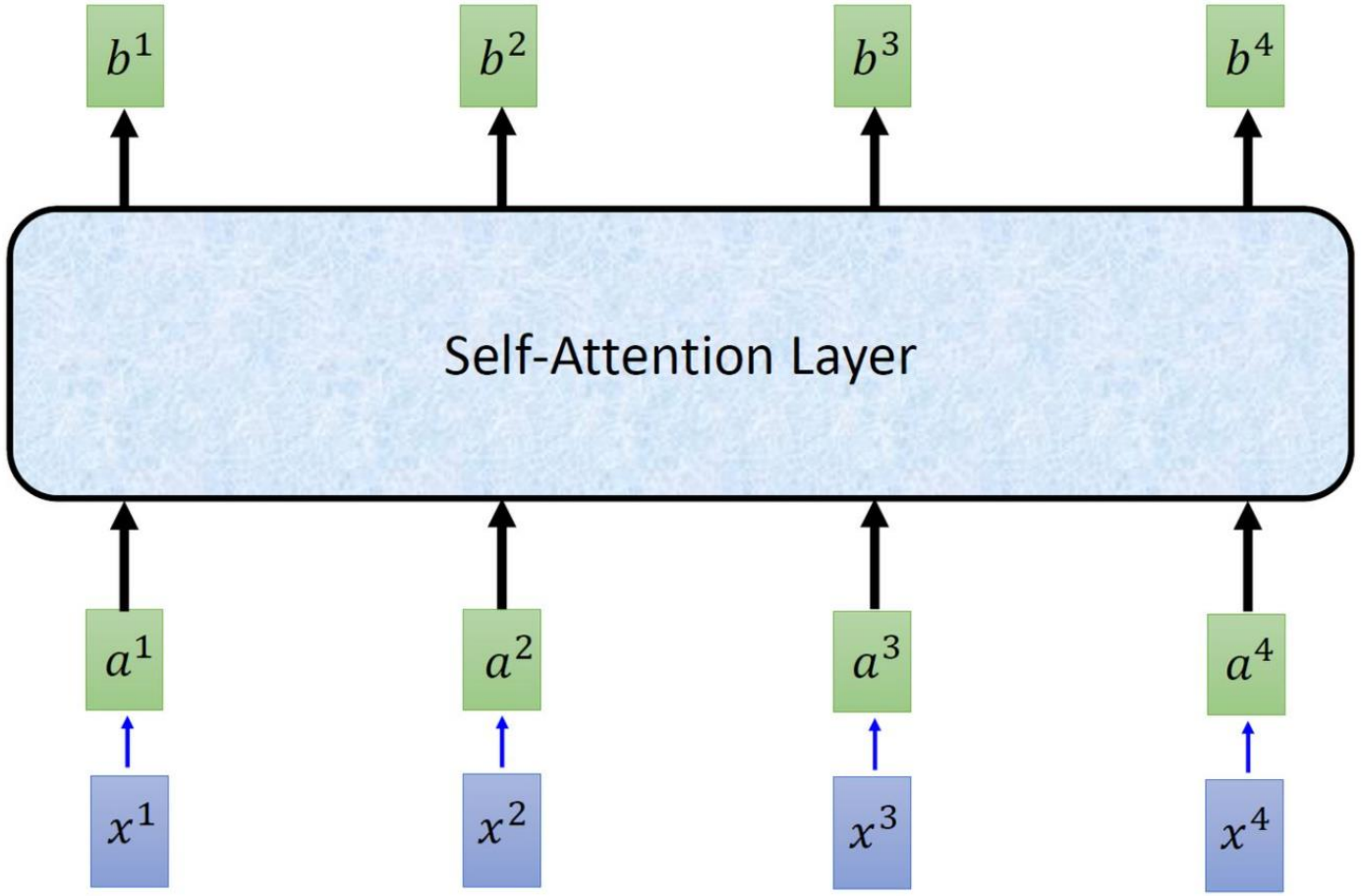
Self-attention

$$\hat{\alpha}_{1,i} = exp(\alpha_{1,i}) / \sum_j exp(\alpha_{1,j})$$



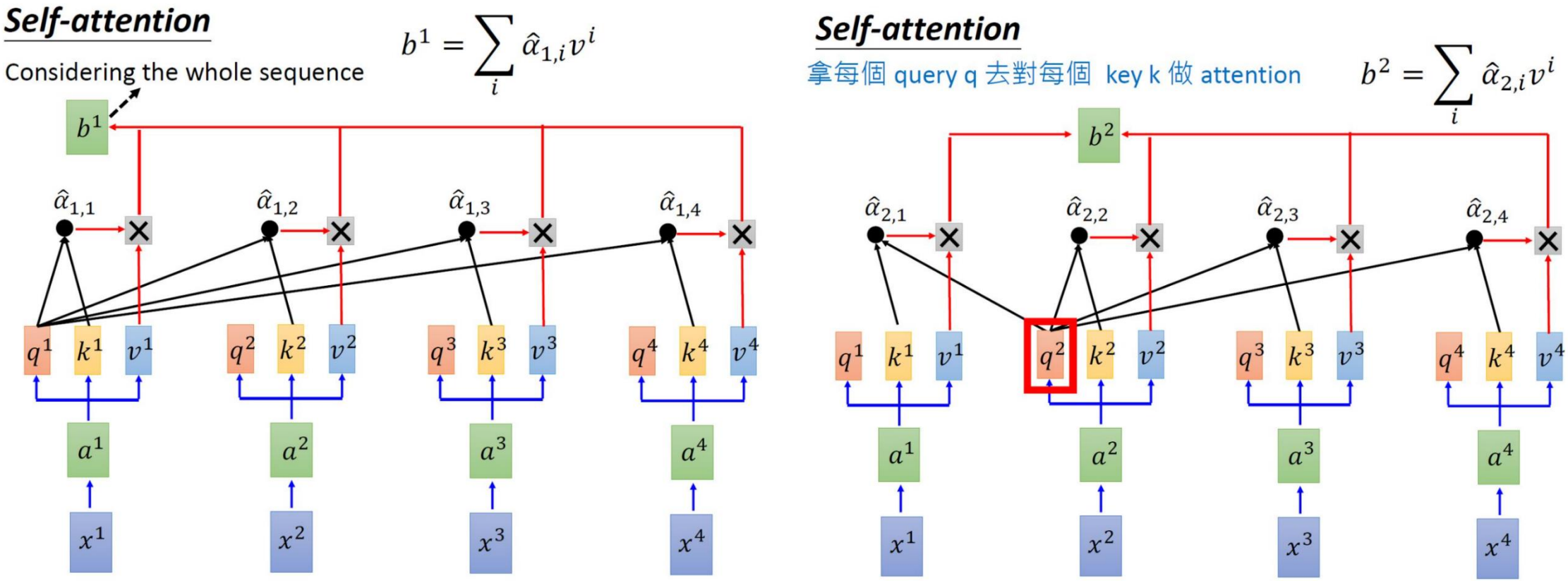
Self-Attention

- 输出$b^1, b^2, \ldots$



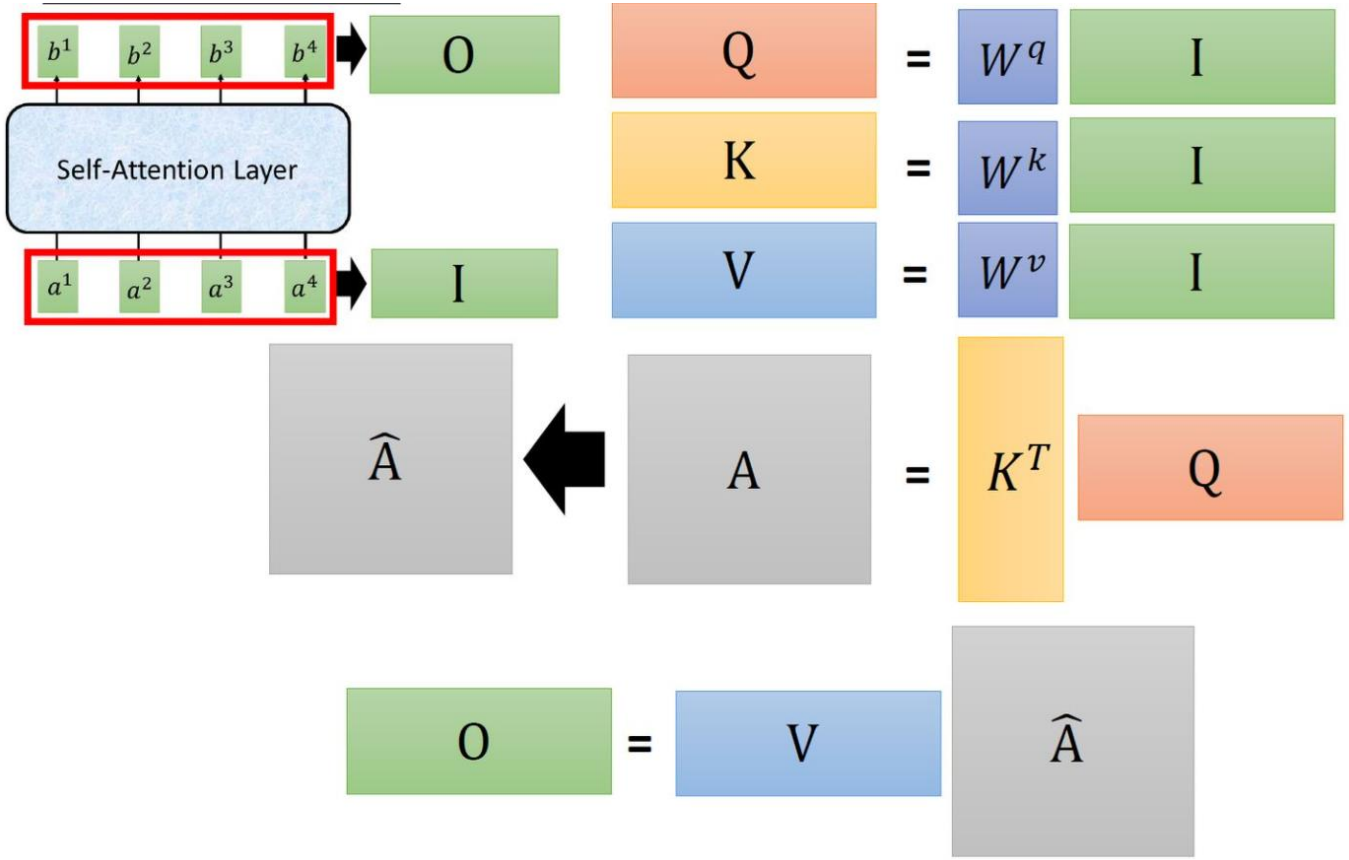
Self-Attention

- 输出$b^1, b^2, \ldots$



Self-Attention

- 矩阵形式
  - 并行计算



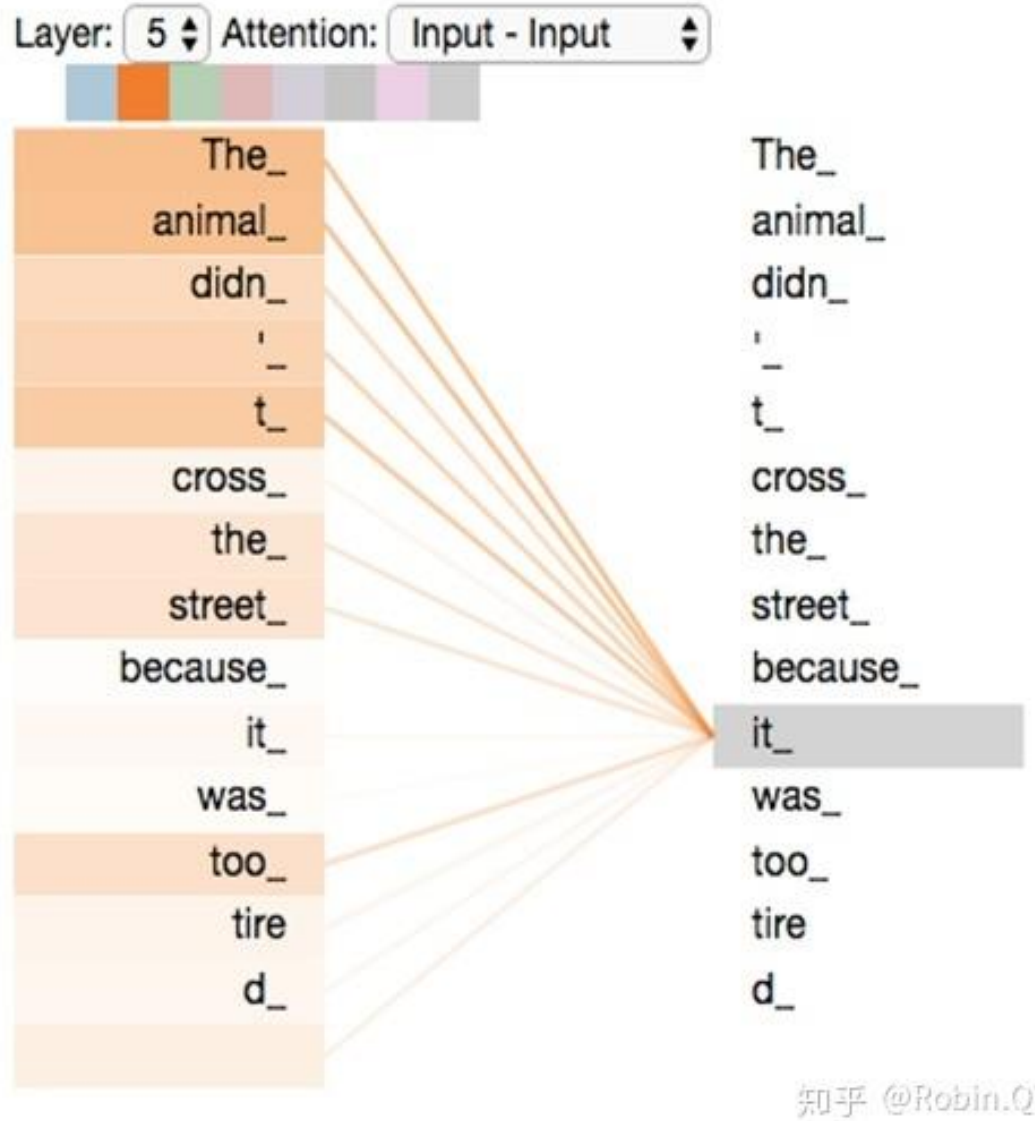
$$\underbrace{q^1\,q^2\,q^3\,q^4}_{Q} = W^q \underbrace{a^1\,a^2\,a^3\,a^4}_{I}$$

$$\underbrace{k^1\,k^2\,k^3\,k^4}_{K} = W^k \underbrace{a^1\,a^2\,a^3\,a^4}_{I}$$

$$\underbrace{v^1\,v^2\,v^3\,v^4}_{V} = W^v \underbrace{a^1\,a^2\,a^3\,a^4}_{I}$$

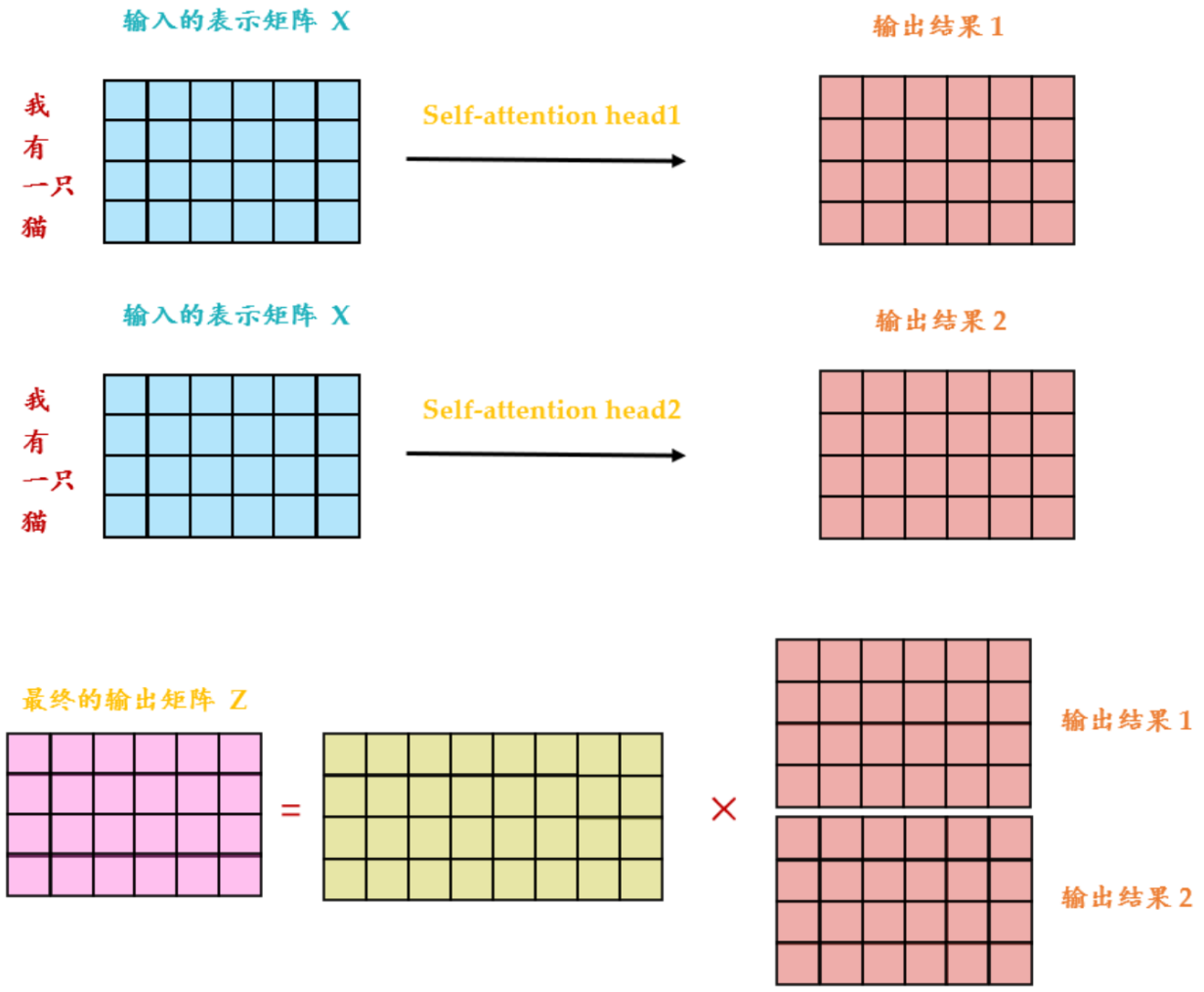
Self-Attention

- 什么效果?
  - Attention
  - 通过上下文来表示每个token



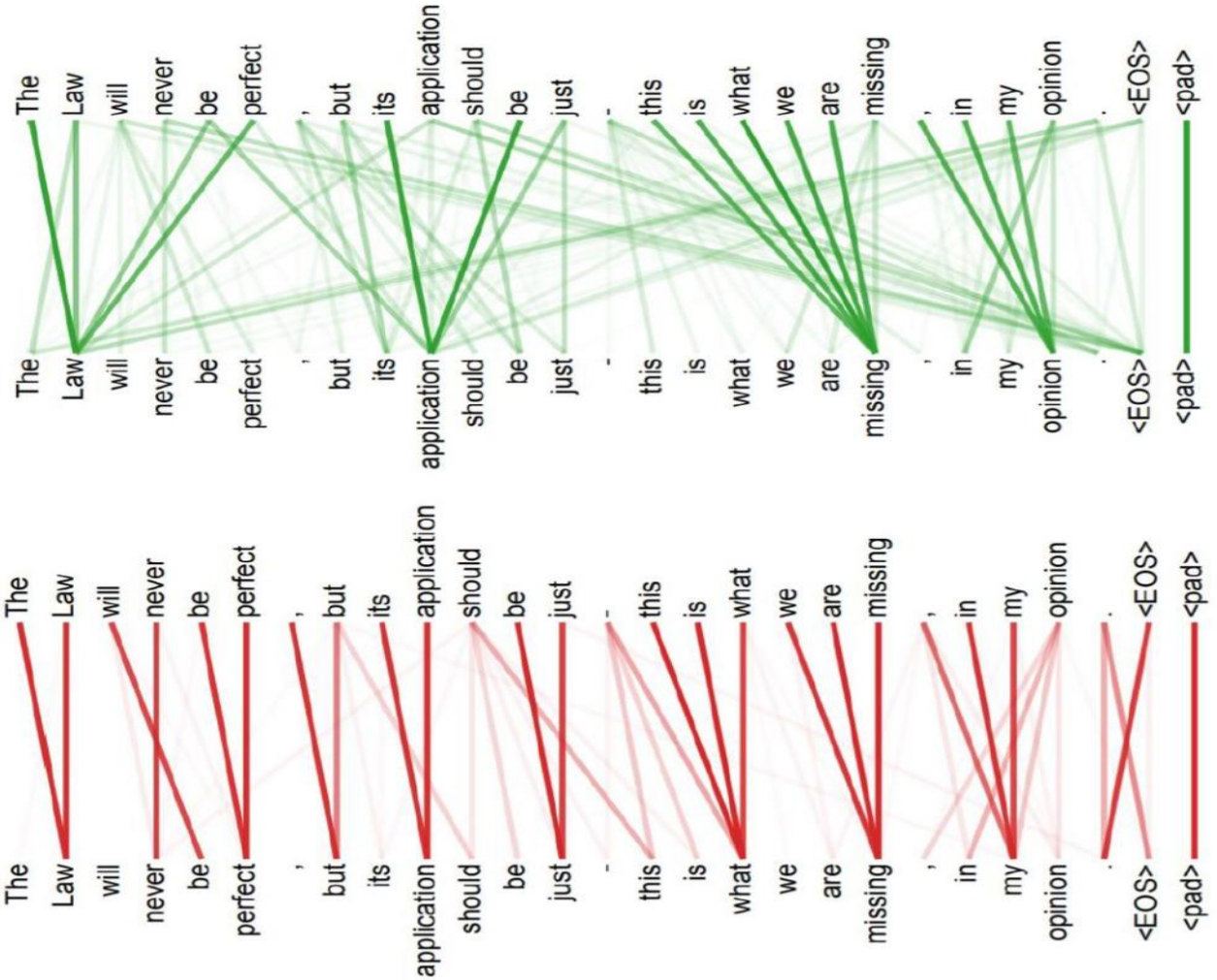
Multi-head Self-Attention

- 多套SA模块
  - 输出Concat.
  - 传入一个Linear层得到最终的输出



Multi-head Self-Attention

- 多套SA模块
  - 输出Concat.
  - 传入一个Linear层
得到最终的输出



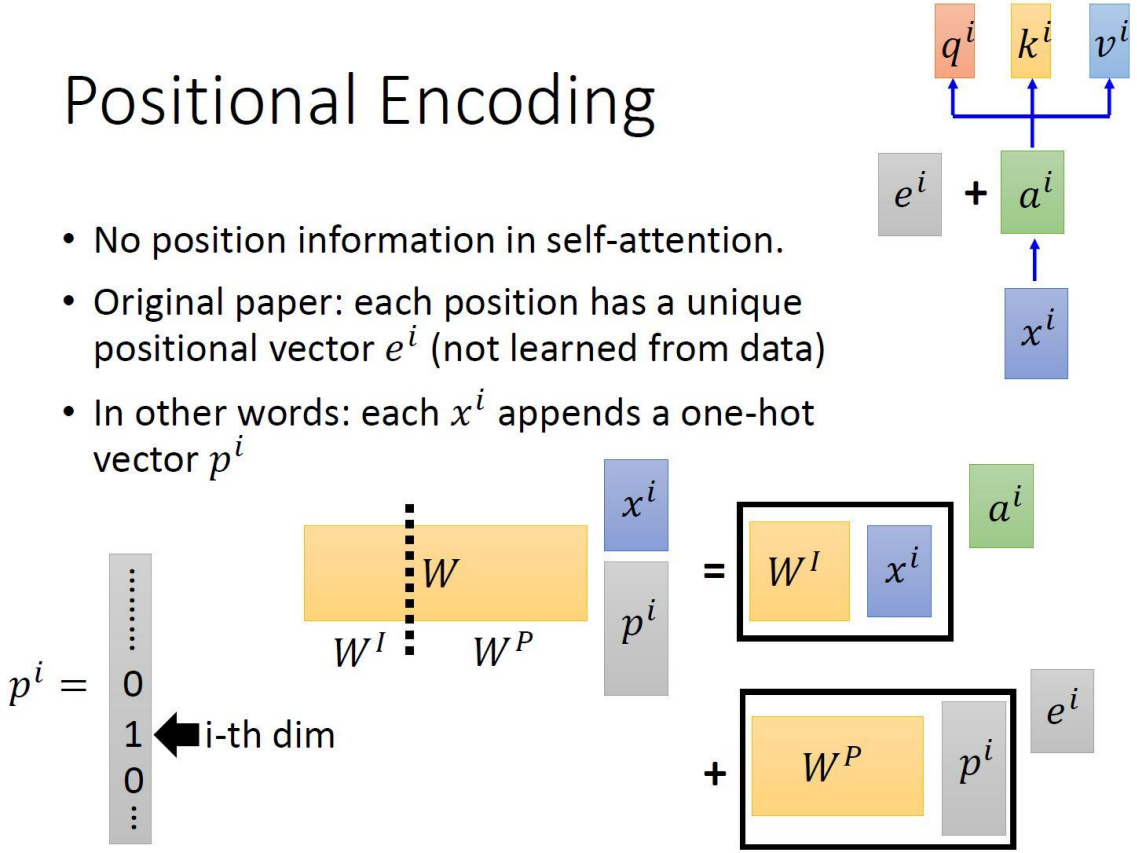
- Why?
  - 不同head关注不同信息
  - 右例
    - 上关注global信息
    - 下关注local信息

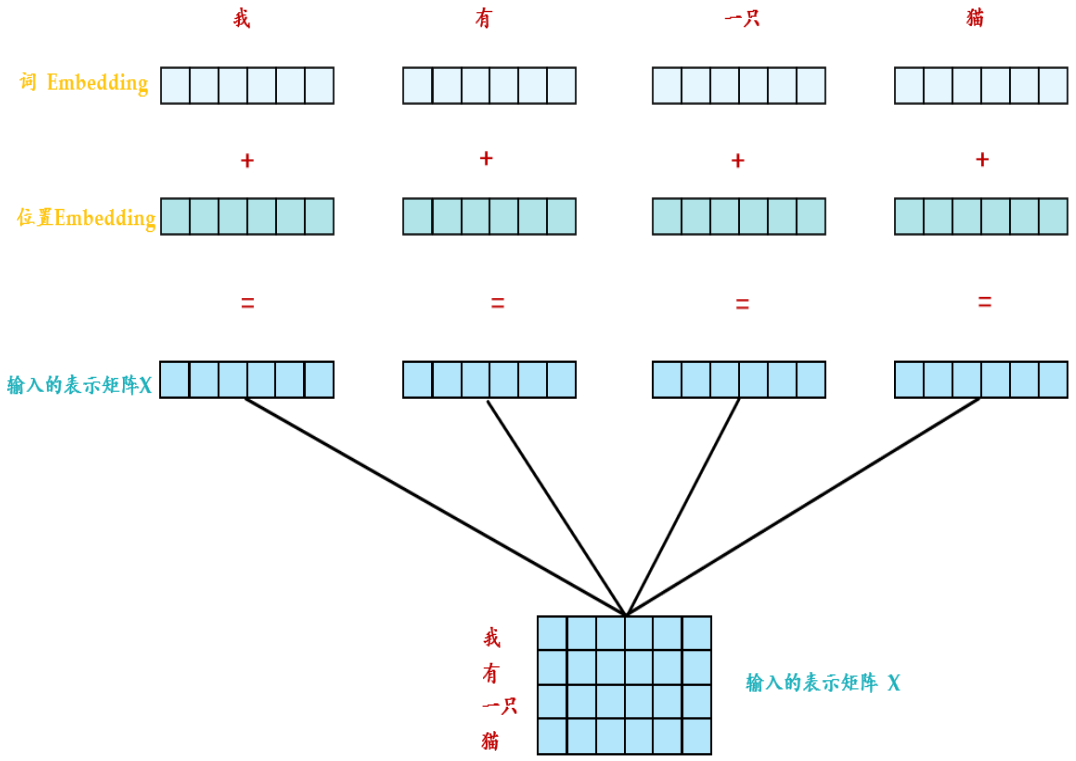
Self-Attention：位置信息

- 序列中Token的位置信息不能丢！

Positional Encoding

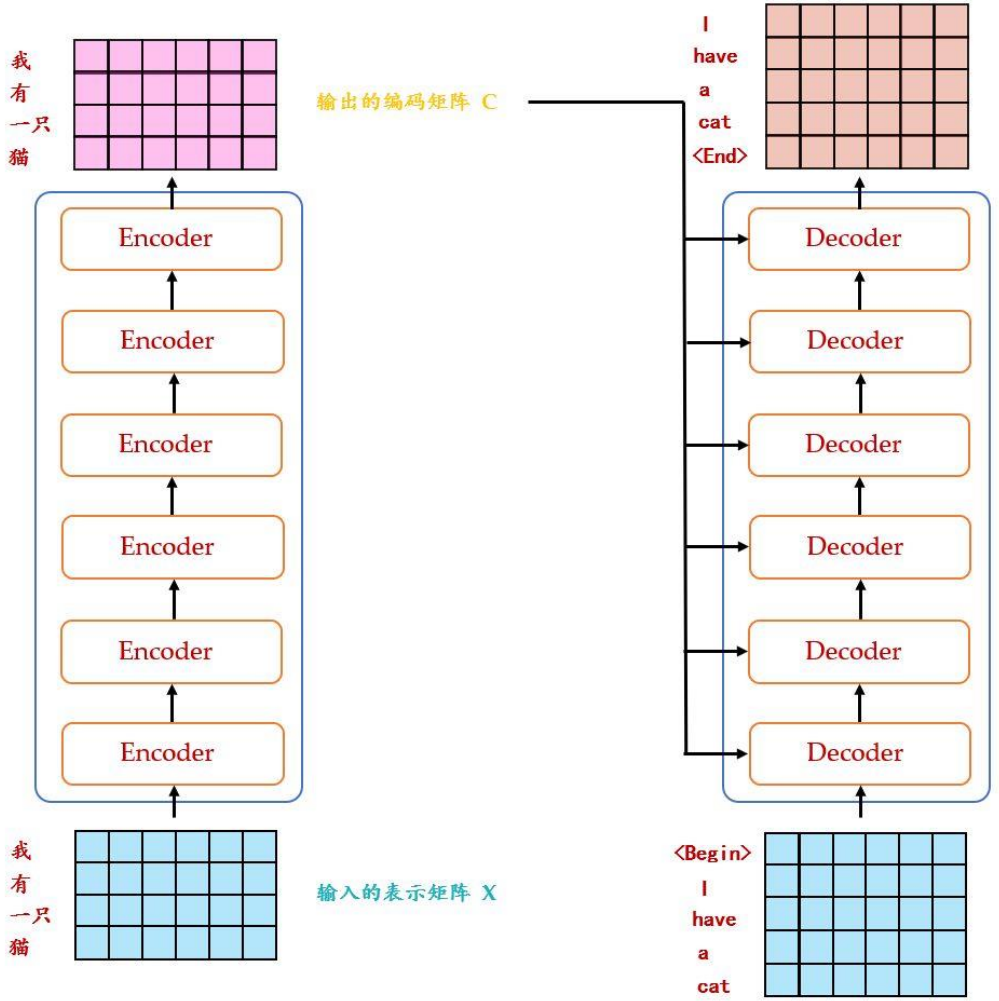
- No position information in self-attention.
- Original paper: each position has a unique positional vector e^i (not learned from data)
- In other words: each x^i appends a one-hot vector p^i





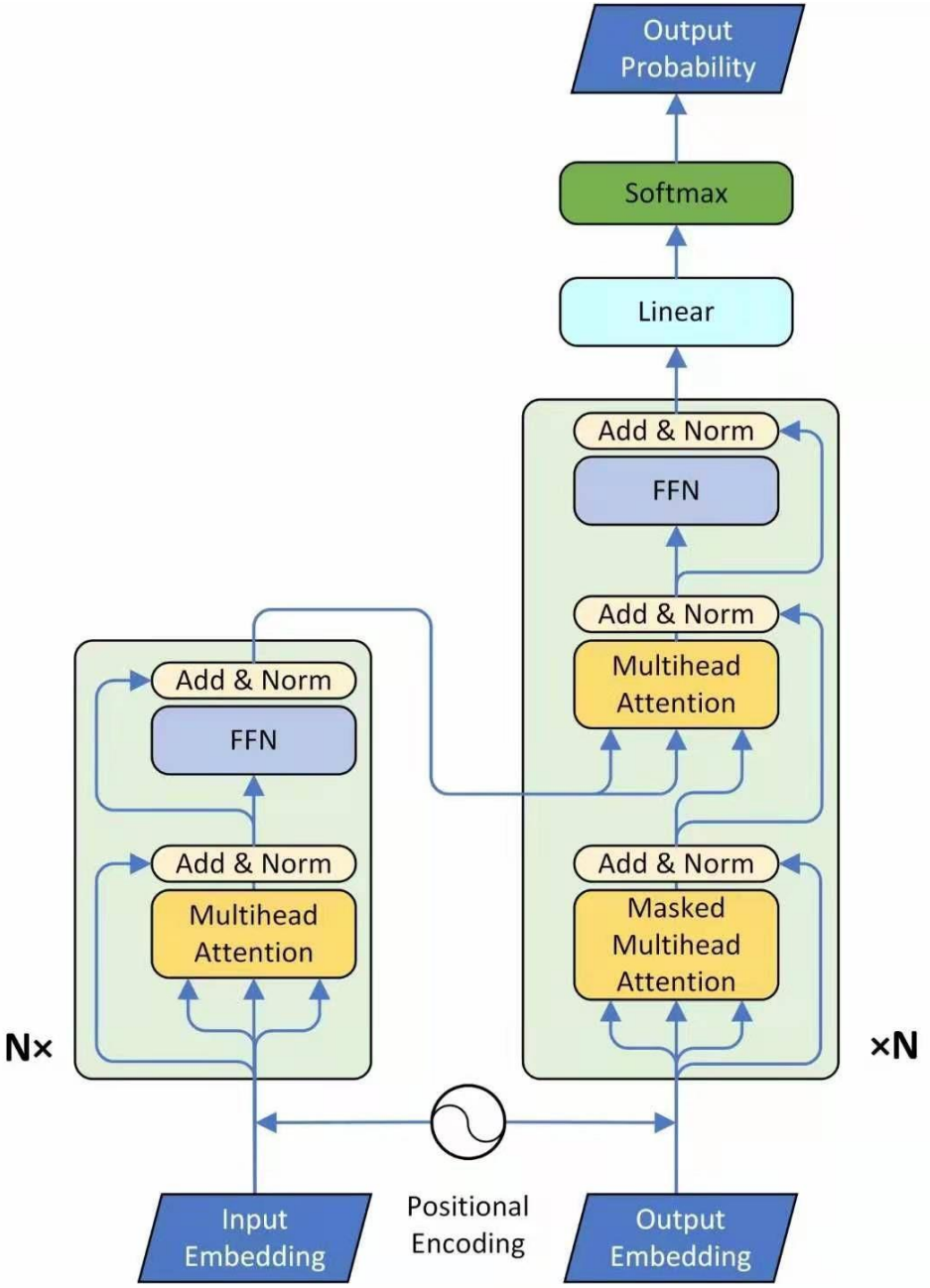
再看Transformer

- 左侧为 Encoder block，右侧为 Decoder block
- 黄色圈中的部分为Multi-Head Attention，是由多个Self-Attention组成的
- Encoder block 包含一个 Multi-Head Attention; Decoder block 包含两个 Multi-Head Attention (其中有一个用到 Masked)
- Multi-Head Attention 上方还包括一个 Add & Norm 层，Add 表示残差连接 (Residual Connection)，Norm 表示 Layer Normalization
- 比如说在Encoder Input处的输入是机器学习，在Decoder Input处的输入是<BOS>，输出是machine。再下一个时刻在Decoder Input处的输入是machine，输出是learning。不断重复直到输出是句点(.)代表翻译结束。



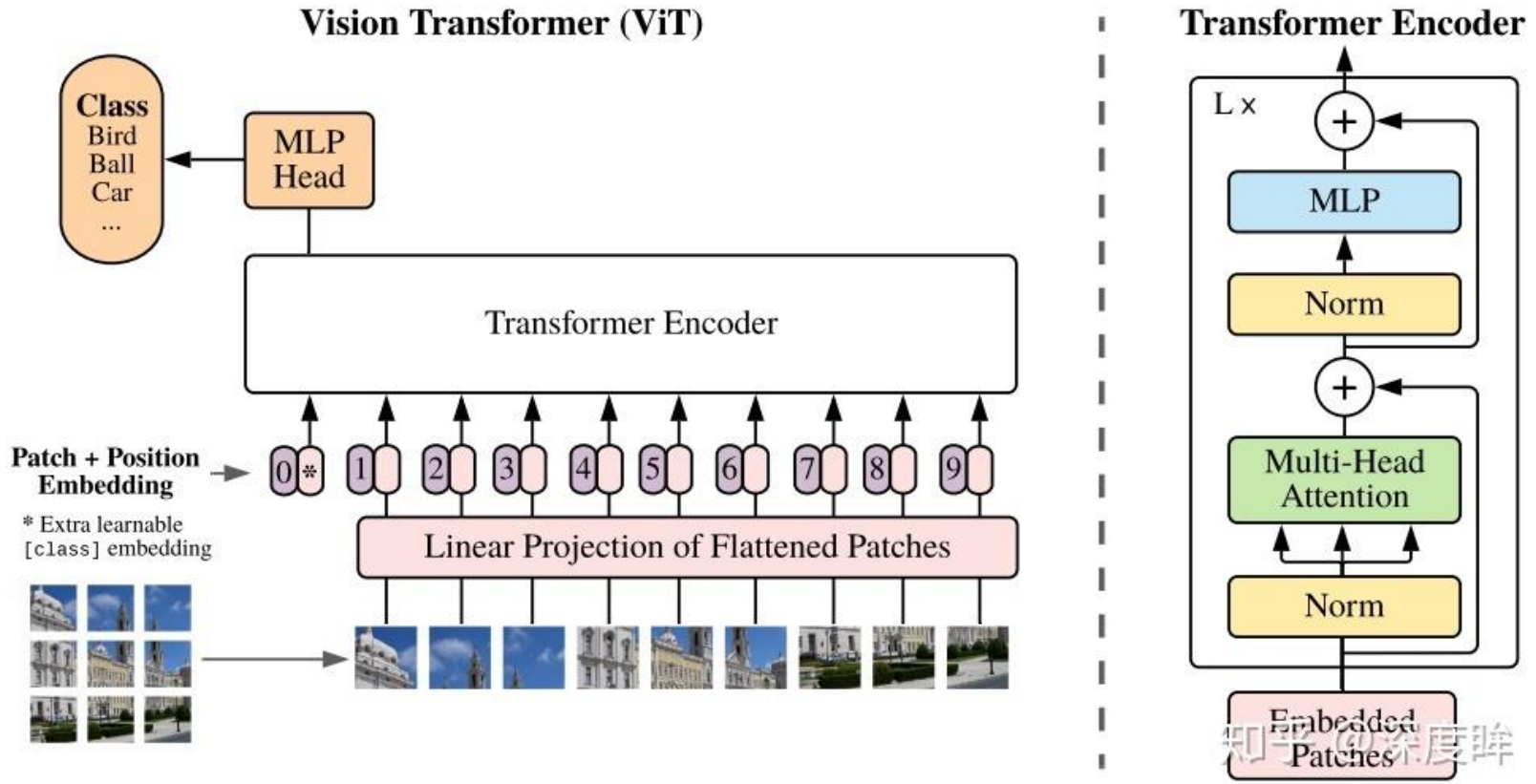
再看Transformer

- 左侧为 Encoder block，右侧为 Decoder block
- 黄色圈中的部分为Multi-Head Attention，是由多个Self-Attention组成的
- Encoder block 包含一个 Multi-Head Attention; Decoder block 包含两个 Multi-Head Attention (其中有一个用到 Masked)
- Multi-Head Attention 上方还包括一个 Add & Norm 层，Add 表示残差连接 (Residual Connection)，Norm 表示 Layer Normalization
- 比如说在Encoder Input处的输入是机器学习，在Decoder Input处的输入是<BOS>，输出是machine。再下一个时刻在Decoder Input处的输入是machine，输出是learning。不断重复直到输出是句点(.)代表翻译结束。



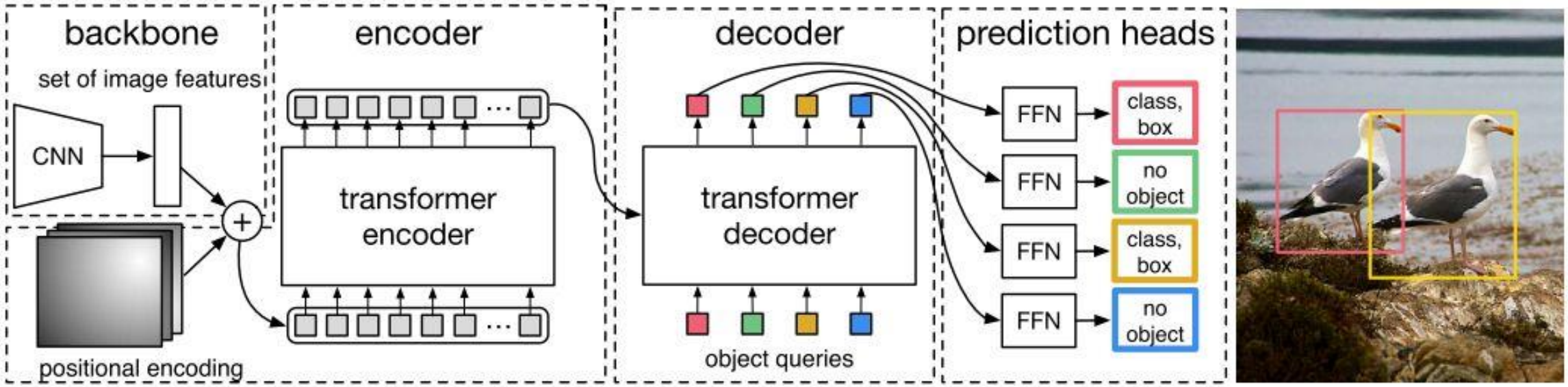
Vision Transformer

- Vision is language!
  - ViT for Image Classification



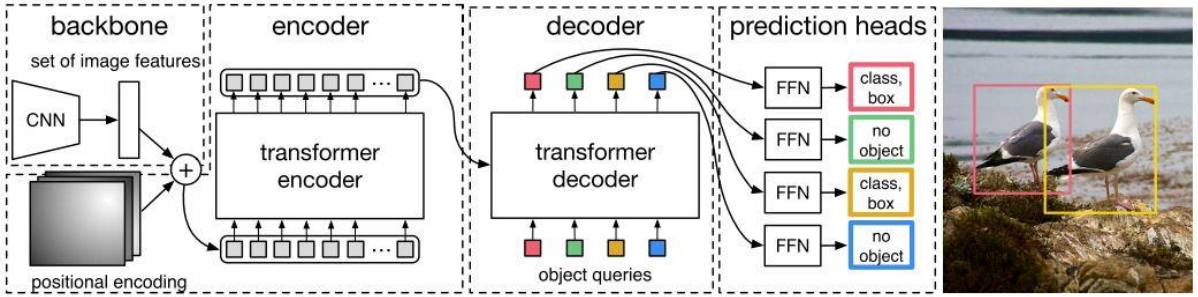
Vision Transformer

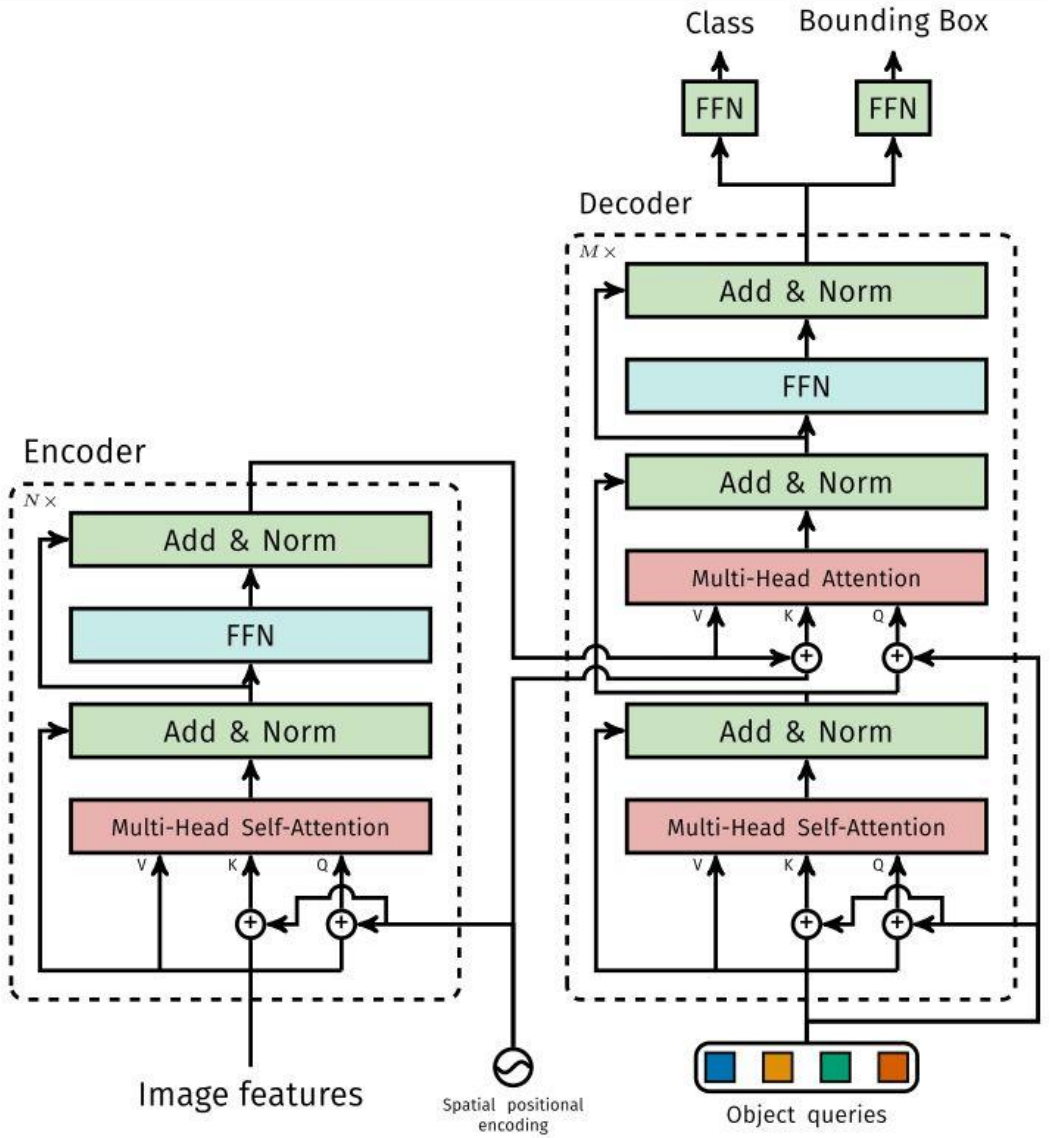
- Vision is language!
  - DETR for Object Detection



Vision Transformer

- Vision is language!
  - DETR for Object Detection





谢谢！