西安交通大学 XI'AN JIAOTONG UNIVERSITY | IAIR Est. 1986 Institute of Artificial Intelligence and Robotics | 人工智能学院 College of Artificial Intelligence, XJTU

《高级机器学习》第八章

深度注意力模型

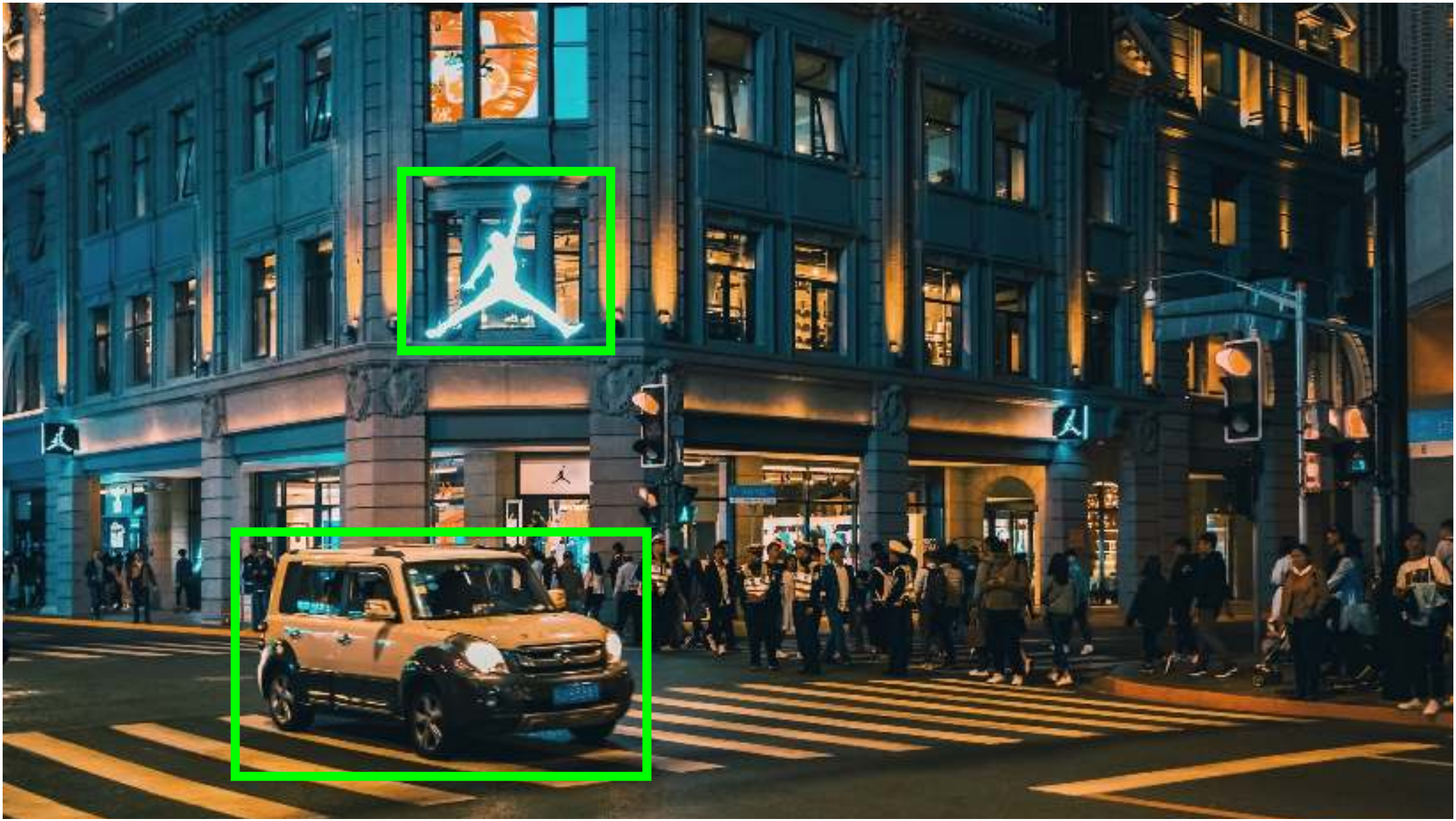
魏平
西安交通大学人工智能学院
人工智能与机器人研究所

IAIR Est. 1986 Institute of Artificial Intelligence and Robotics

人工智能学院 College of Artificial Intelligence, XJTU

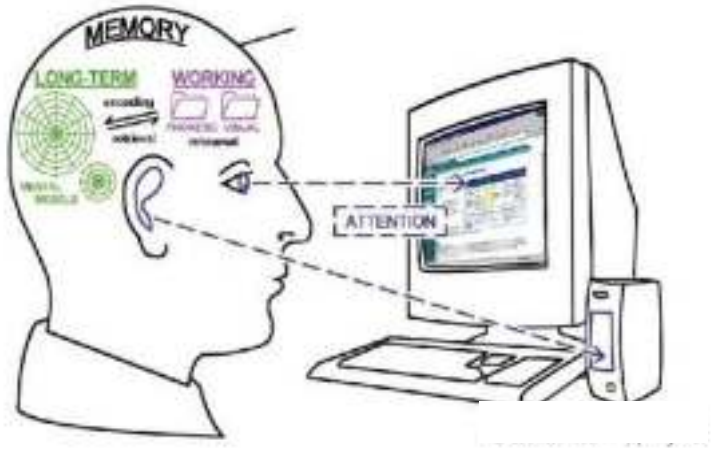
CONTENTS

- 注意力简介
- Transformer 模型
- 视觉Transformer
- 大语言模型

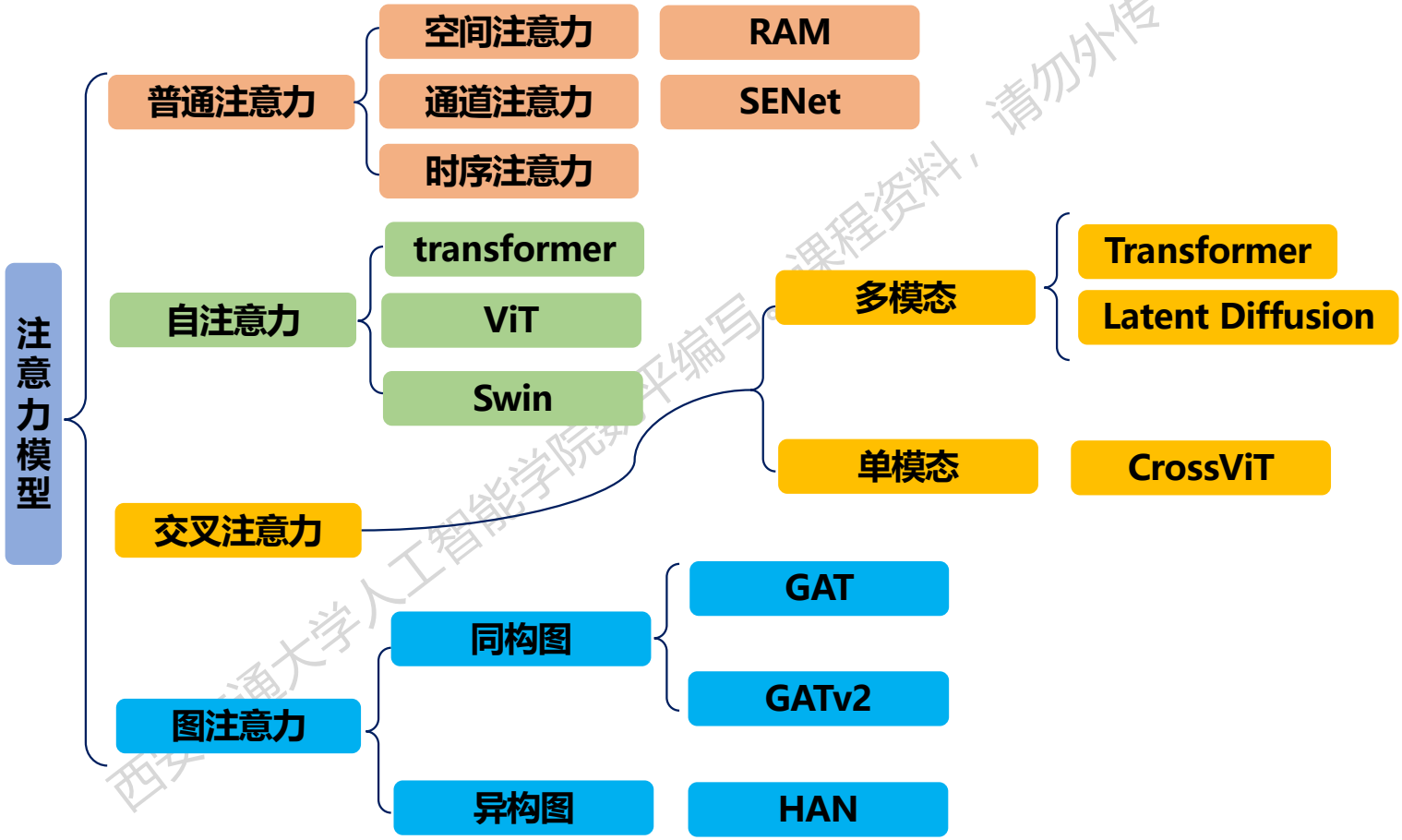


注意力 Attention

- 注意力的概念原本来自于心理学，指人的心理活动指向和集中于某种事物的能力。注意力的本质是从关注全部到关注重点
- 注意力、记忆力、观察力、想象力、思维力一般被称为智力的五个基本因素。注意力被称为人类心灵的门户



注意力模型分类



CONTENTS

- 注意力简介
- Transformer 模型
- 视觉Transformer
- 大语言模型

Transformer 模型引入

- 2017 年，Google 机器翻译团队发表《Attention is All You Need》提出Transformer，采用Attention机制来进行机器翻译任务，取得了很好的效果，注意力机制也成为了近期的研究热点

Attention Is All You Need

Ashish Vaswani[1]
Google Brain
avaswani@google.com

Noam Shazeer[1]
Google Brain
noam@google.com

Niki Parmar[1]
Google Research
nikip@google.com

Jakob Uszkoreit[1]
Google Research
usz@google.com

Llion Jones[1]
Google Research
llion@google.com

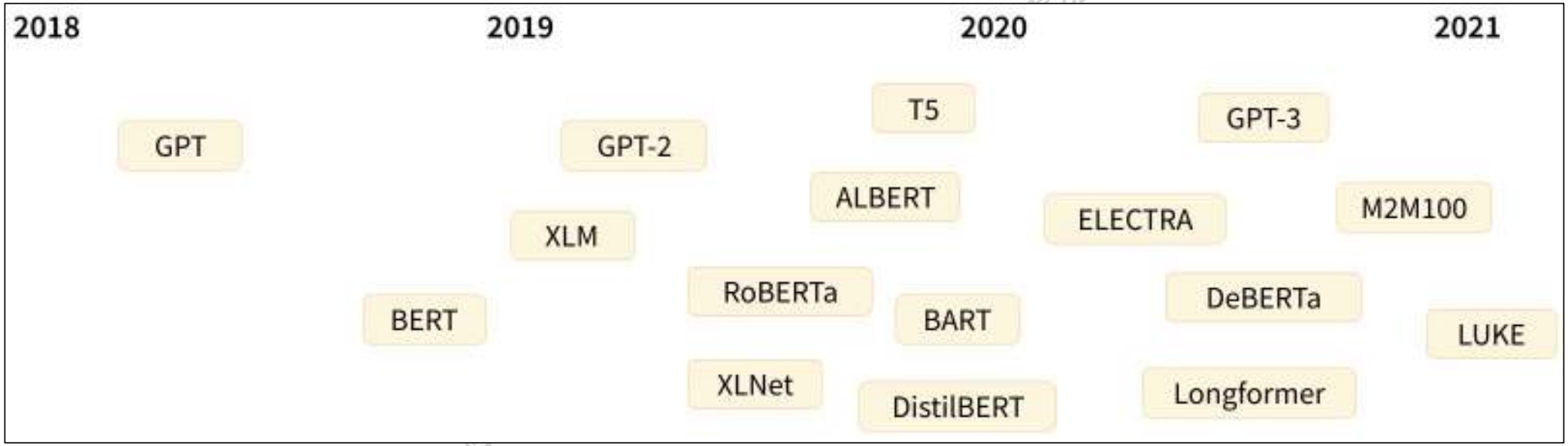
Aidan N. Gomez[1]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser[1]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[1]
illia.polosukhin@gmail.com

Transformer 模型发展

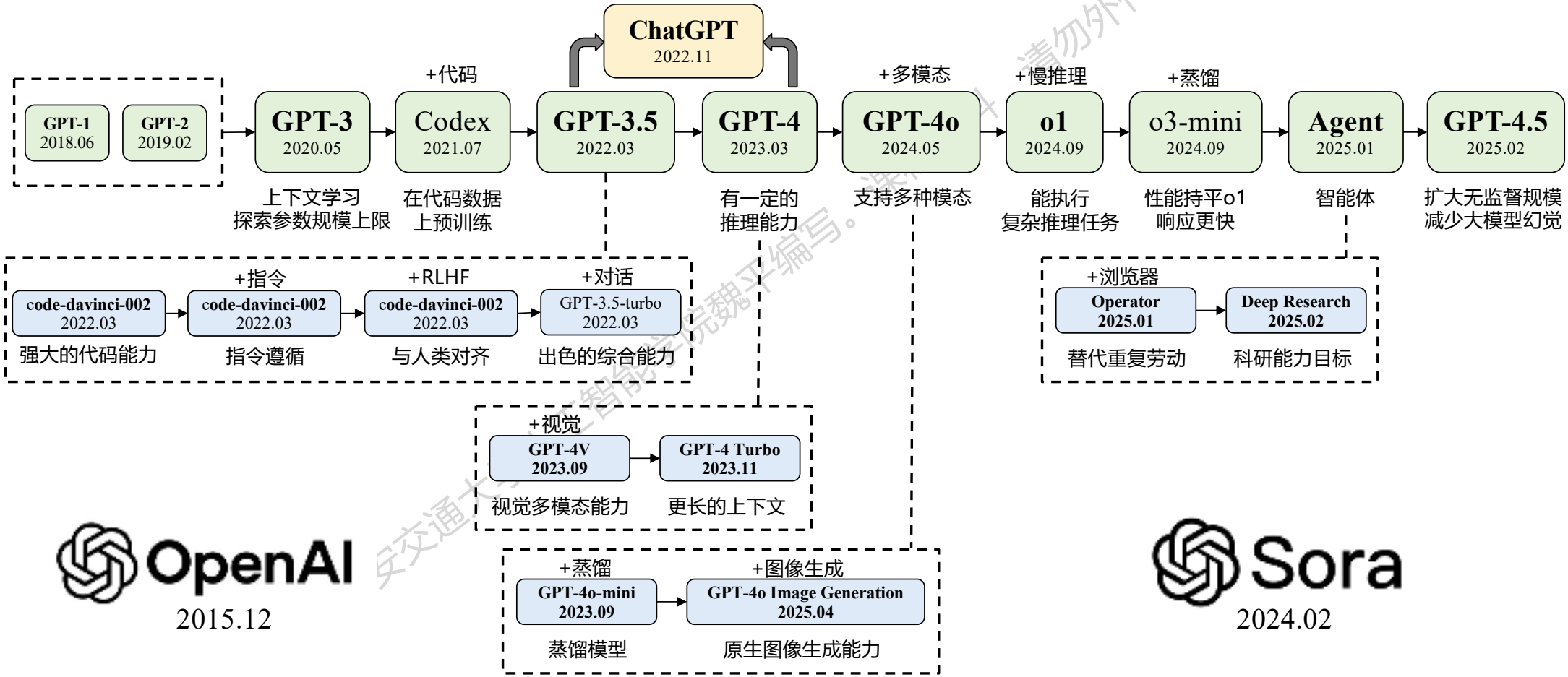
- Transformer具有出色的并行计算和长序列处理能力，极大促进了AI模型的发展



GPT: Generative Pretrained Transformer
BERT: Bidirectional Encoder Representations from Transformers

Transformer 与大模型

GPT: Generative Pre-trained Transformer

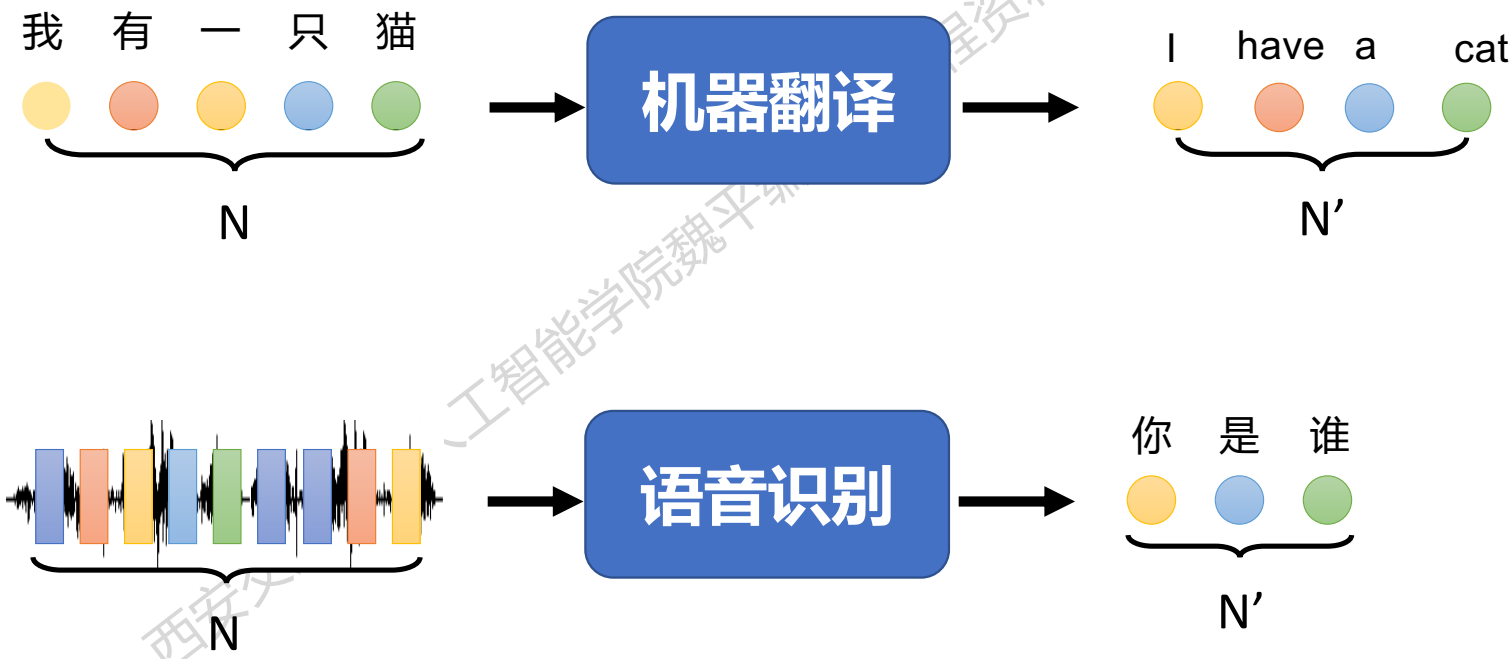


OpenAI
2015.12

Sora
2024.02

Seq2Seq任务 1

- Seq2Seq指的是输入和输出是序列的任务，输入和输出的长度不定

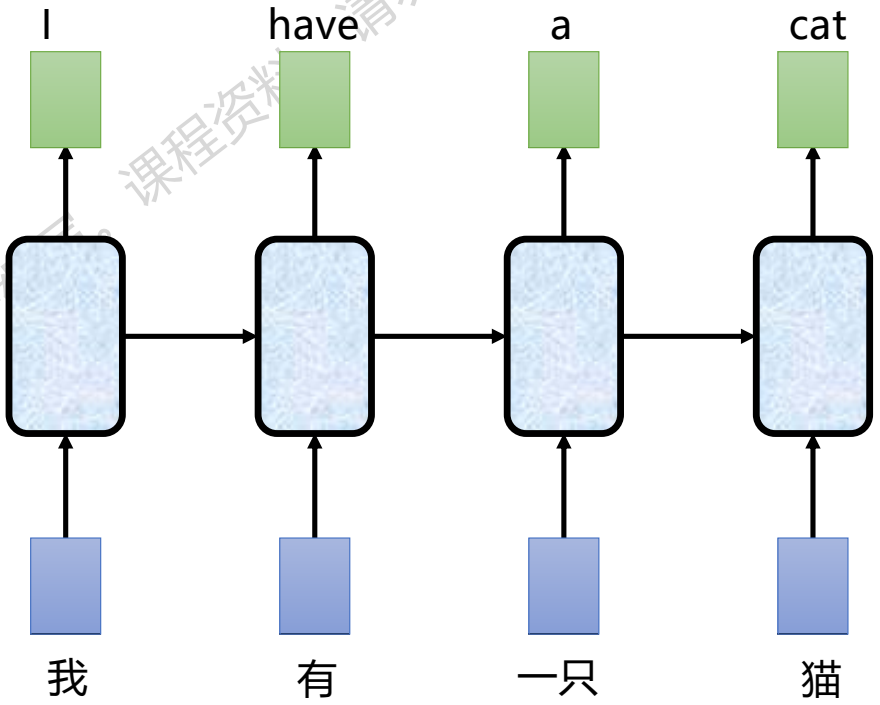


Seq2Seq任务 2

RNN结构

例：机器翻译任务

输入语言：我有一只猫
输出语言：I have a cat

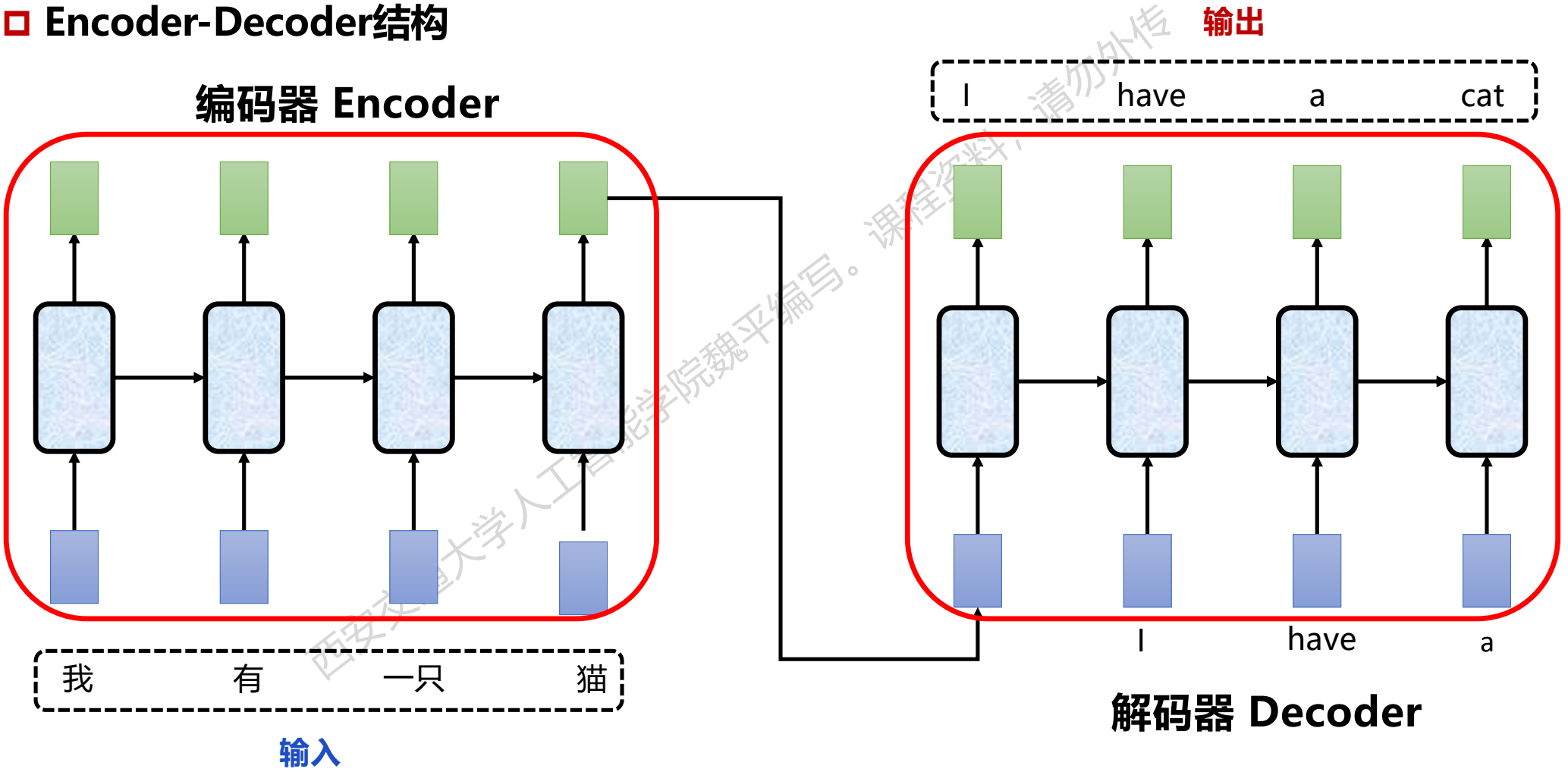


RNN结构缺陷

- 不能并行计算
- 超长期记忆效果差

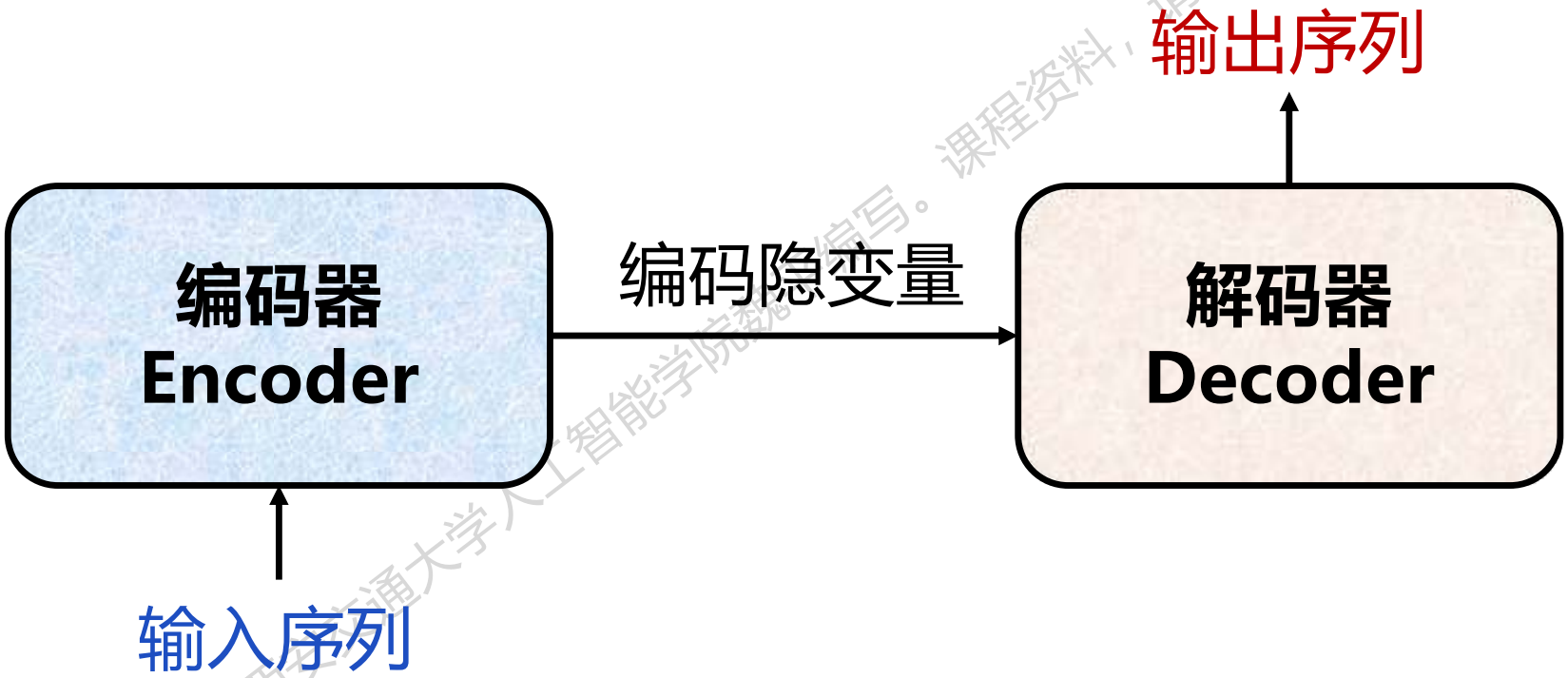
Seq2Seq任务 3

- Encoder-Decoder结构



Seq2Seq任务 4

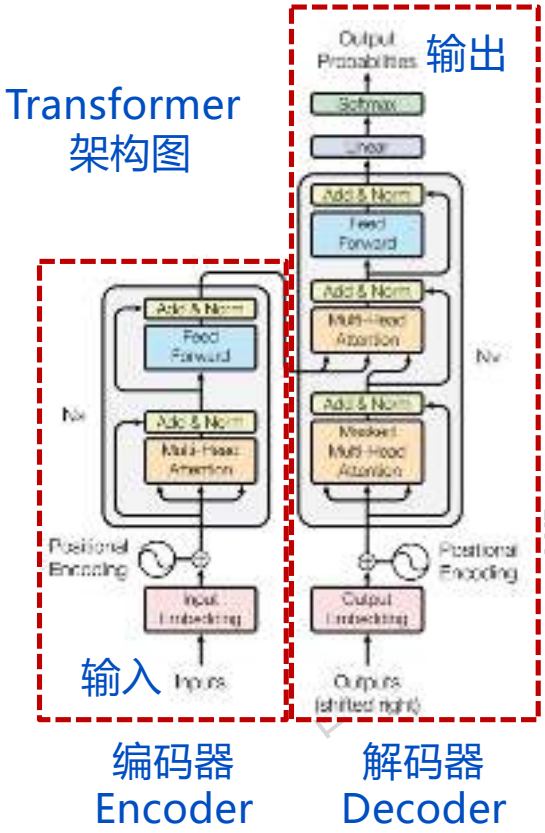
- Encoder-Decoder结构



Transformer 整体架构 1

- Transformer 是一种神经网络架构，具有出色的并行计算能力和长序列处理能力

  Ashish Vaswani, et. al, Attention is all you need, NIPS, 2017



三大关键技术

自注意力机制

计算元素间关联强度，全局依赖建模

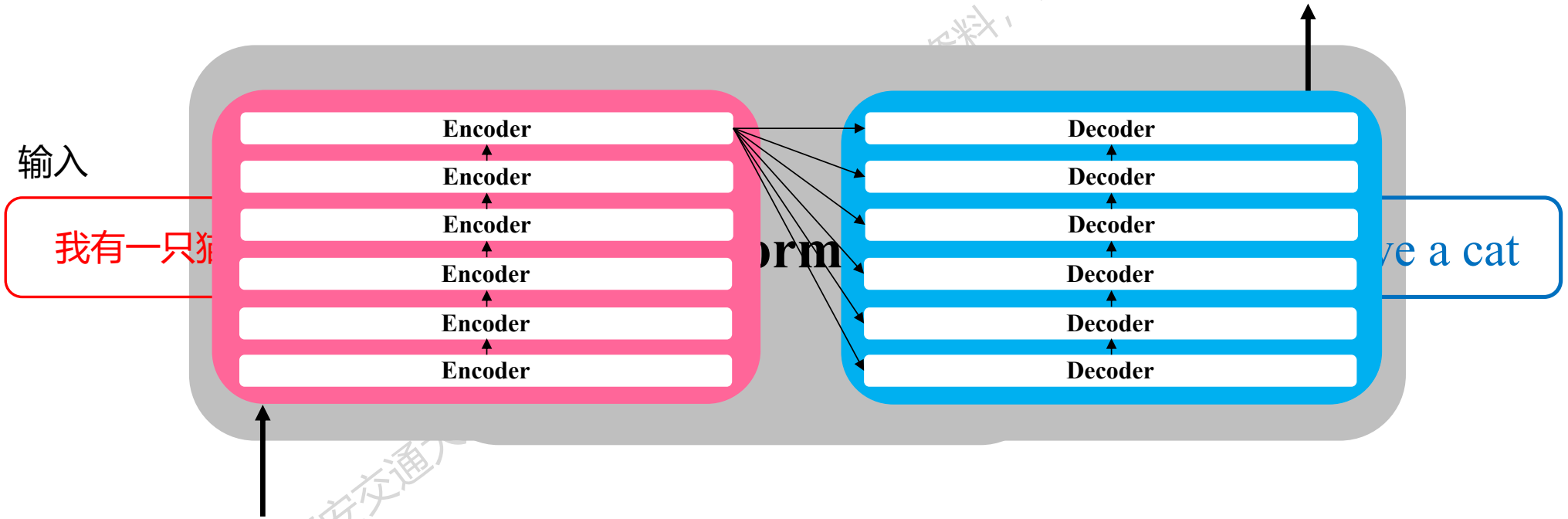
多头注意力

并行运行多组自注意力机制

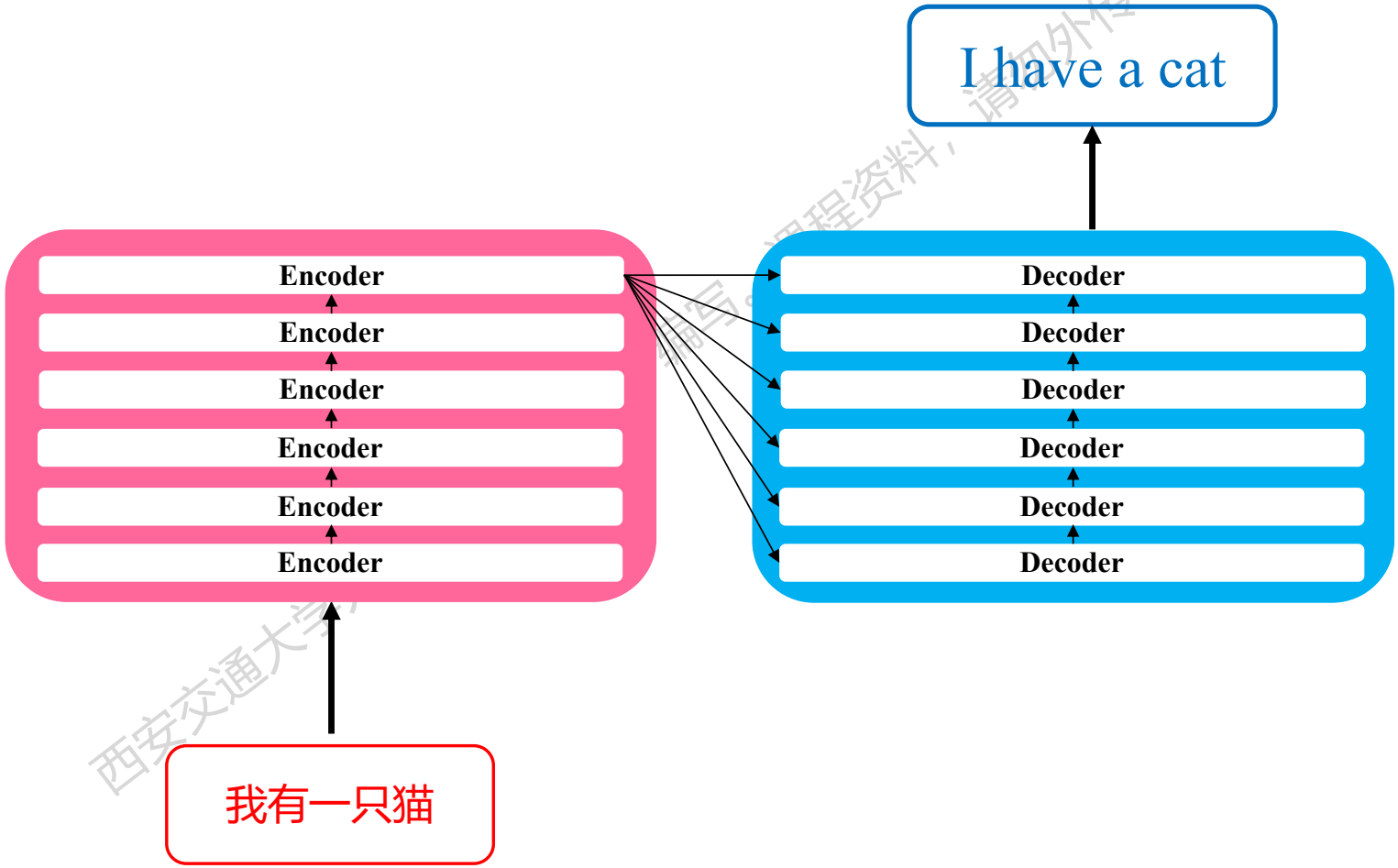
位置编码

显式注入位置信息

Transformer 整体架构 2

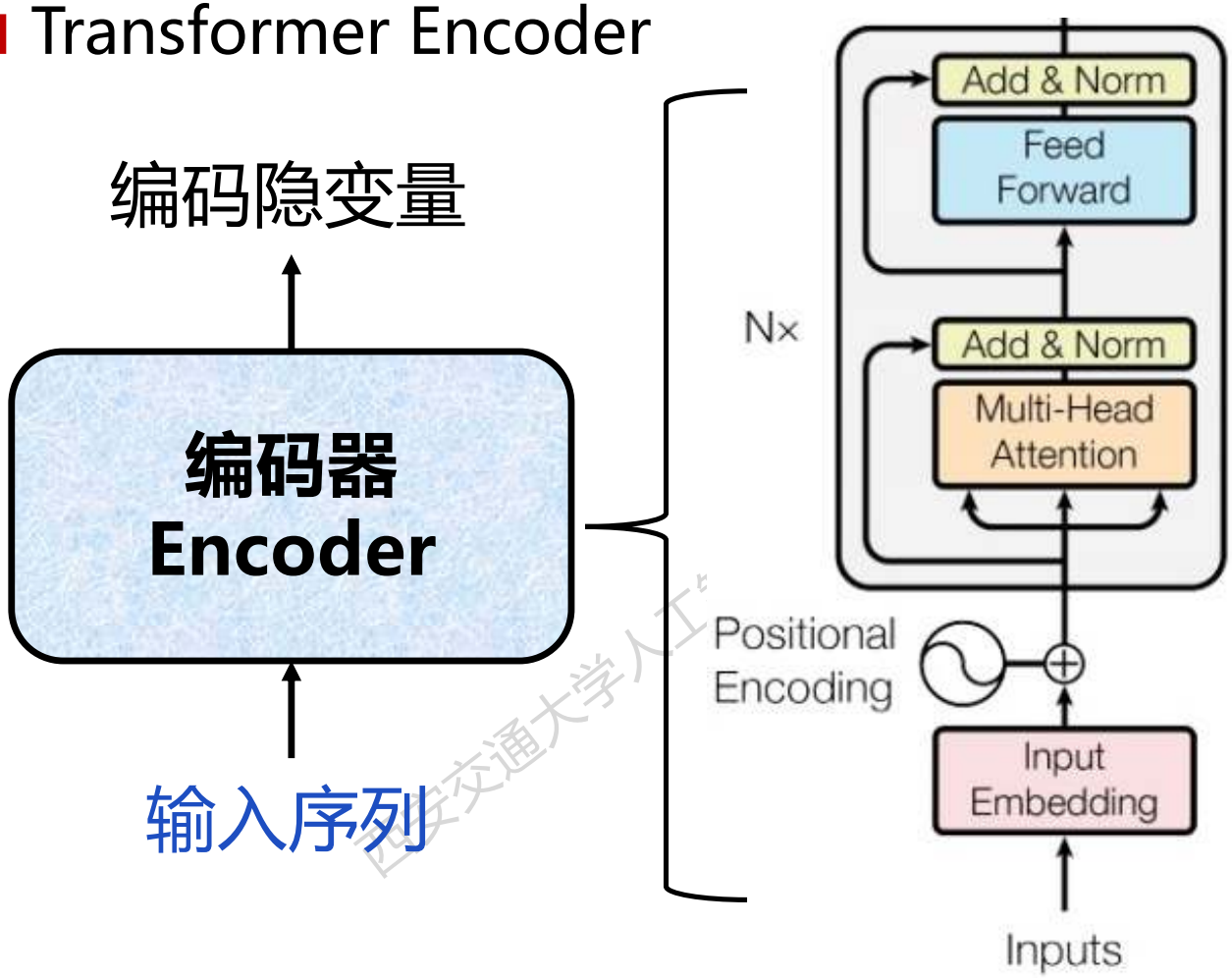


Transformer 整体架构 3



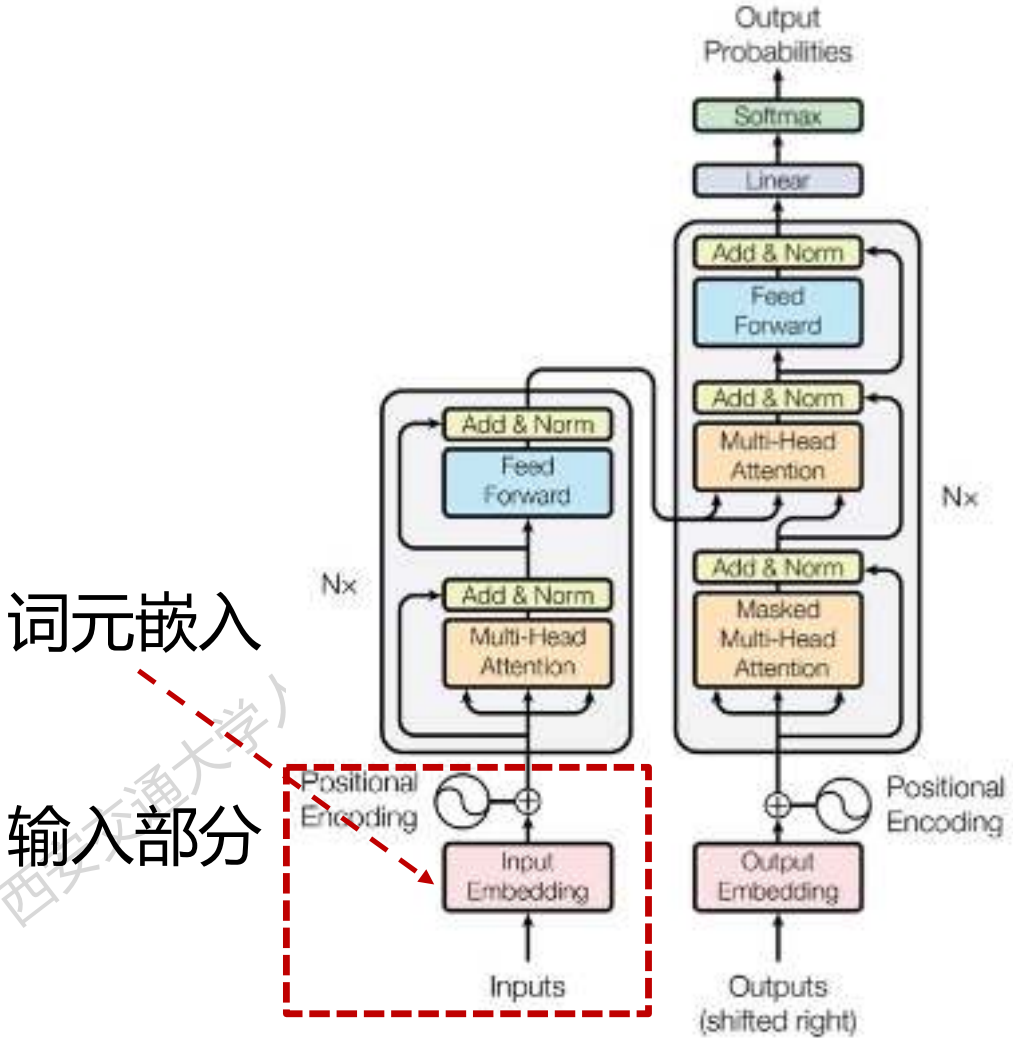
编码器 Encoder

- Transformer Encoder



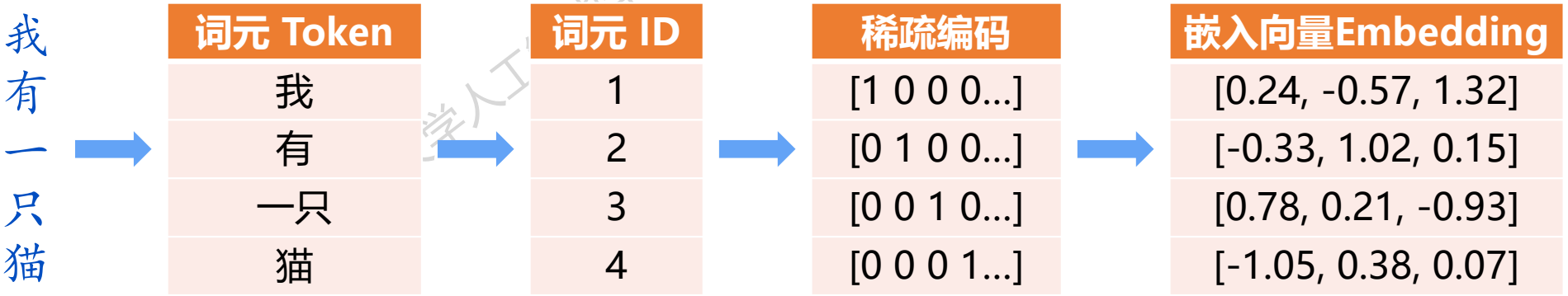
- 编码器的输入是经过嵌入和位置编码处理的词向量序列
- 输出是编码后的上下文表征隐向量
- 编码器由N 个相同的编码器层堆叠而成
- 每个编码器层包含两个核心子层：多头自注意力机制和前馈神经网络
- 每个子层之间采用残差连接，并进行层归一化

输入部分1



输入部分1: 词元嵌入 Token Embedding

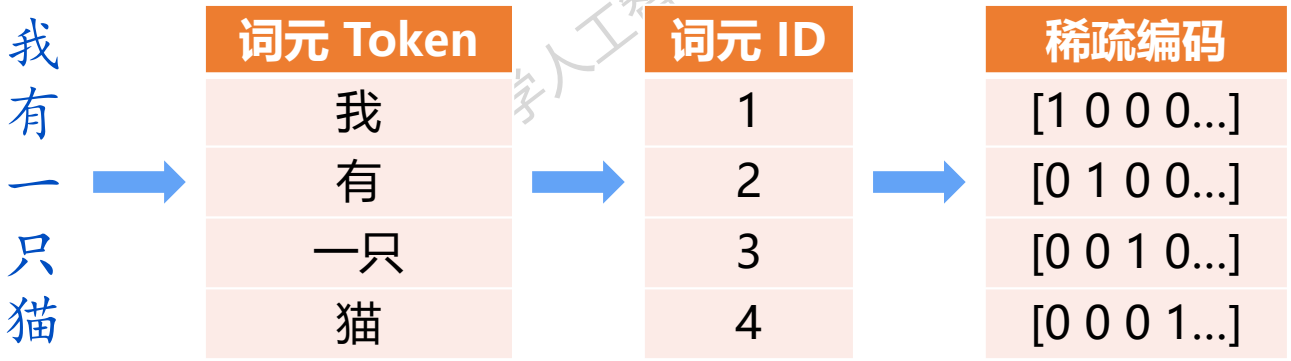
- 在输入处理步骤中，词元嵌入（Token Embedding)将离散的文本数据转换为模型可以理解的固定维度的数值向量，往往是语言模型的第一步处理；词元嵌入由分词(tokenization)和嵌入(embedding)两个环节构成
- 分词(tokenization)将输入文本分解为若干个词元(token)，并将每个词元转换为数字ID
- 嵌入(embedding)将每个词元映射到一个固定长度的数值向量，用来表示词元的语义特征



输入部分2: 分词 Tokenization

- 分词(tokenization)将输入文本转化为数字ID。具体过程包含将输入文本分解为若干个词元(token)，将每个词元映射为数字ID，将数字ID编码为one-hot稀疏向量；词元映射为数字ID的过程是可逆的，即可以通过ID解码出词元
- 词元是模型处理文本的基本单元，包括4个类型
  - 单词：单个单词，如cat, running
  - 子词：词干和前后缀等，run, ing
  - 字符：单个字符，c, a, t
  - 符号：标记、符号等，[SEP]，<PAD>
- 稀疏编码采用one-hot方式

例：若词表大小为30k，则ID 4 词元稀疏编码向量第4位为1，其余为0



常用的分词方法有 WordPiece 、Sentence-Piece 、BPE等

输入部分3: 嵌入 Embedding

- 分词后将词元的one-hot稀疏编码经嵌入向量空间矩阵映射为稠密的固定维度的编码向量，称为嵌入(embedding)。映射后的向量是连续数值，便于计算；可有效表达词元之间的语义关系，如“猫”、“狗”、“车”；同时也起到语义压缩的作用
- 嵌入向量空间矩阵通过学习得到。静态嵌入向量方法，同一个词在不同上下文中具有相同的向量，Word2Vec、GloVe；上下文动态向量方法，同一个词在不同的上下文中可能有不同的向量表示，可捕捉多义性和语境变化，如BERT、GPT

词元稀疏向量

我	1	0	0	0	0	0
有	0	1	0	0	0	0
一只	0	0	1	0	0	0
猫	0	0	0	1	0	0

4×6

×

嵌入向量空间矩阵

0.24	-0.57	1.32
-0.33	1.02	0.15
0.78	0.21	-0.93
-1.05	0.38	0.07
0.83	0.56	1.45
0.21	0.34	0.75

6×3

=

词元嵌入向量

0.24	-0.57	1.32
-0.33	1.02	0.15
0.78	0.21	-0.93
-1.05	0.38	0.07

4×3

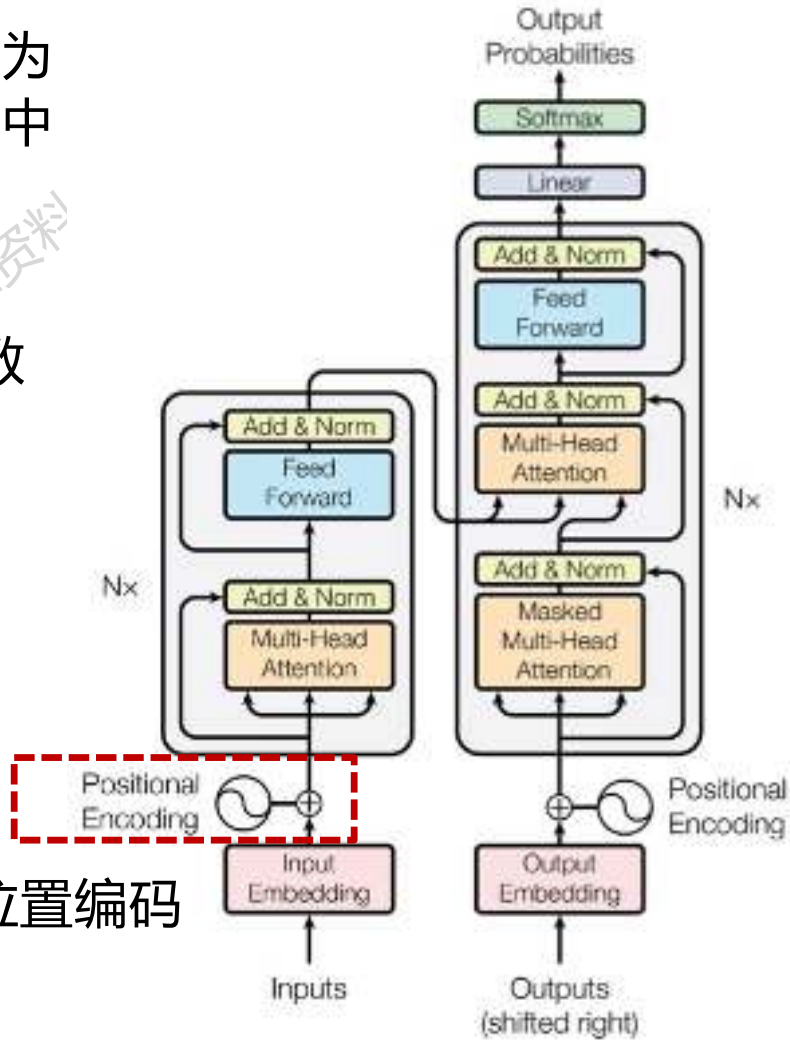
输入部分4: 位置编码

- 相同的单词以不同的顺序排列可能表示不同的语义，为了在这种情况下区分不同的句子，还需要在单词特征中加入位置信息

  e.g. “我吃了一个苹果” “苹果公司发布了新手机”

- 位置编码（Positional Encoding)用固定维度的连续数值向量表达一个token在序列中的位置
- 正余弦位置
  - t是token在序列中的实际位置（第一个token为1，第二个token为2...）
  - $PE_t \in \mathbb{R}^d$是token的位置向量，PE_t^i表示这个位置向量里的第i各元素

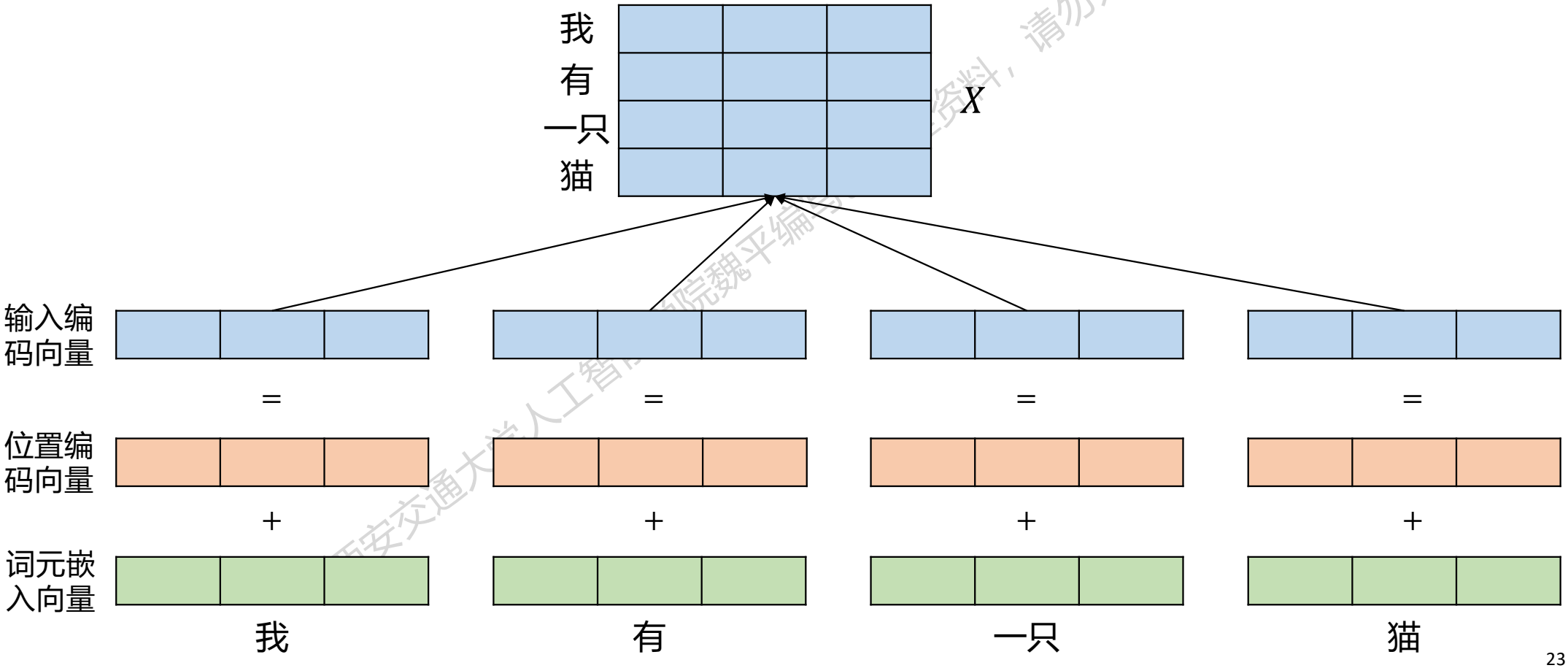
$$PE_t^i = \begin{cases} \sin(\omega_k t), & if\ i = 2k \\ \cos(\omega_k t), & if\ i = 2k+1 \end{cases} \qquad \omega_k = \frac{1}{10000^{2k/d}}$$



位置编码

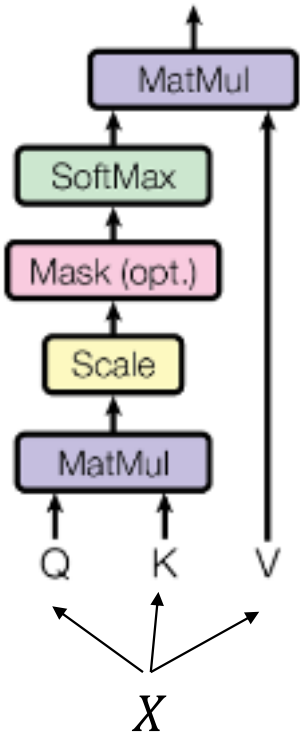
输入部分5: 输入编码

- 输入编码 = 词元嵌入向量 + 位置编码



编码器 Encoder 1：自注意力

- 缩放点积注意力 Scaled Dot-Product Attention
  - 将每个词的特征向量x_i映射到向量$query$查询, key键，$value$值，分别为q_i, k_i, v_i

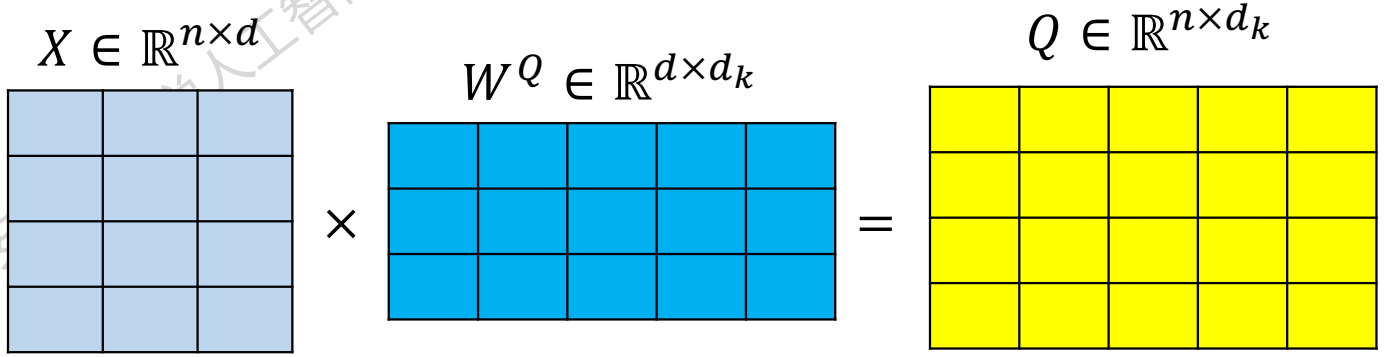


$$q_i = x_i W^Q \quad k_i = x_i W^K \quad v_i = x_i W^V$$

矩阵形式

$$Q = XW^Q \quad K = XW^K \quad V = XW^V$$

$$W^Q \in \mathbb{R}^{d \times d_k}, Q \in \mathbb{R}^{n \times d_k}$$
$$W^K \in \mathbb{R}^{d \times d_k}, K \in \mathbb{R}^{n \times d_k}$$
$$W^V \in \mathbb{R}^{d \times d_v}, V \in \mathbb{R}^{n \times d_v}$$

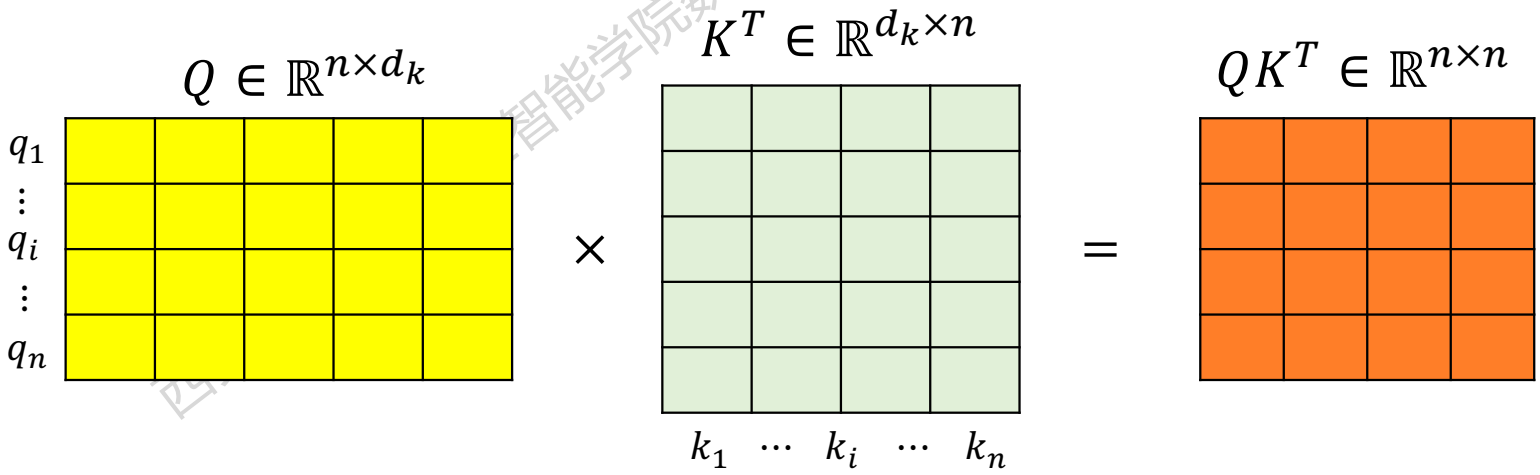


编码器 Encoder 2：自注意力

- 自注意力计算
  - 得到矩阵K，Q，V后，自注意力的输出为

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

softmax(·)对每一行归一化

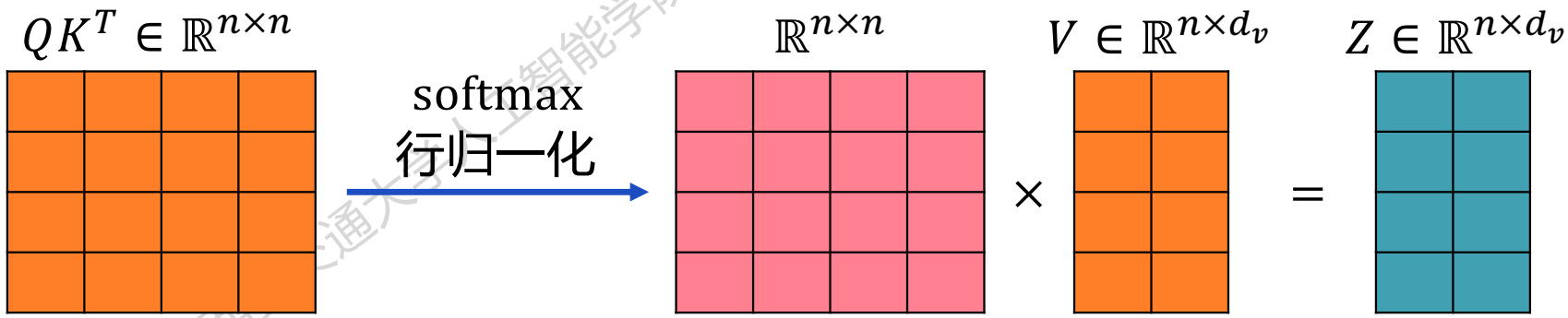


编码器 Encoder 3：自注意力

- 自注意力计算
  - 得到矩阵K，Q，V后，自注意力的输出为

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

softmax(·)对每一行归一化



编码器 Encoder 4：自注意力

- 自注意力将输入变换到另外一个特征空间

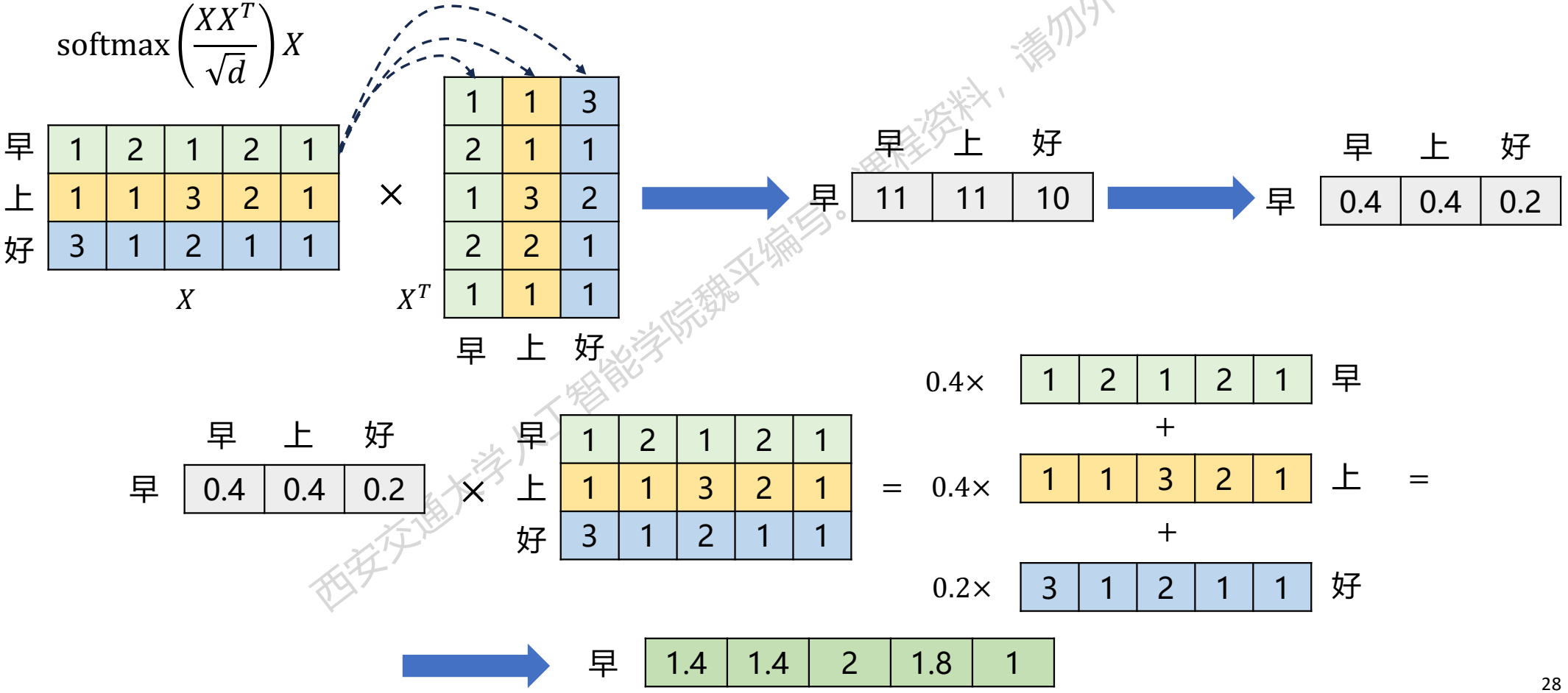
$$Q = XW^Q$$
$$K = XW^K$$
$$V = XW^V$$

$$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

编码器 Encoder 5：自注意力的含义

- 两个向量的内积可以视为衡量两个向量相关度的方式，值越大相关性越高

$$\text{softmax}\left(\frac{XX^T}{\sqrt{d}}\right)X$$



编码器 Encoder 6：K，Q，V的含义

- K, Q, V是X的线性变换，为什么要引入K, Q, V而不直接用X

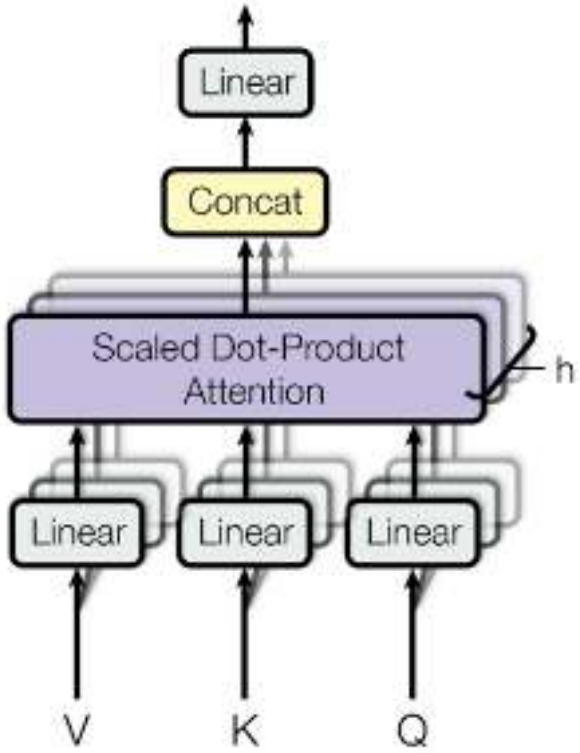
$$Q = XW^Q$$
$$K = XW^K$$
$$V = XW^V$$
$$Z = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$Z = \text{softmax}\left(\frac{XX^T}{\sqrt{d}}\right)X$$

线性变换矩阵W^Q， W^K， W^V都是可学习的参数，引入K, Q, V可以增强模型的表征和拟合能力；同时， W^Q， W^K， W^V也起到对原始输入X的加工和缓冲作用，使得模型更加鲁棒

编码器 Encoder 7：多头注意力

- 为了捕捉不同表征子空间的信息，实际应用中常采用多头注意力机制，即实施多个自注意力机制后合并输出，增强模型的表征能力



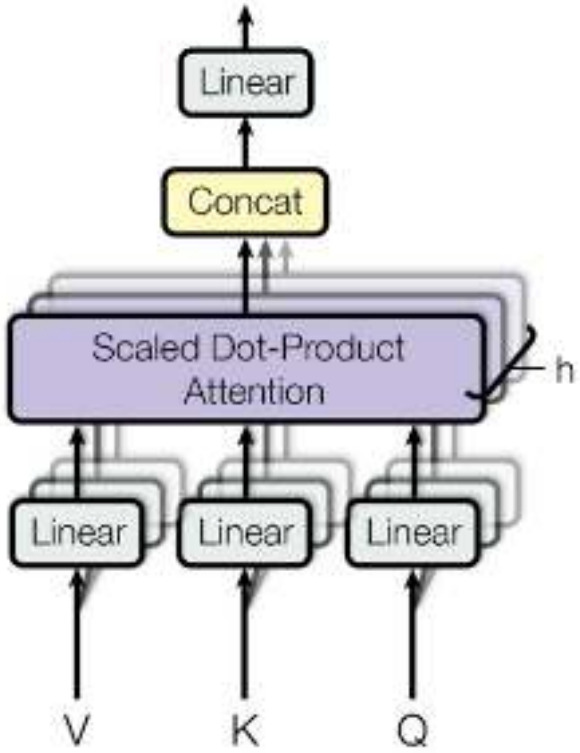
$$K_i = XW_i^K$$
$$Q_i = XW_i^Q$$
$$V_i = XW_i^V$$

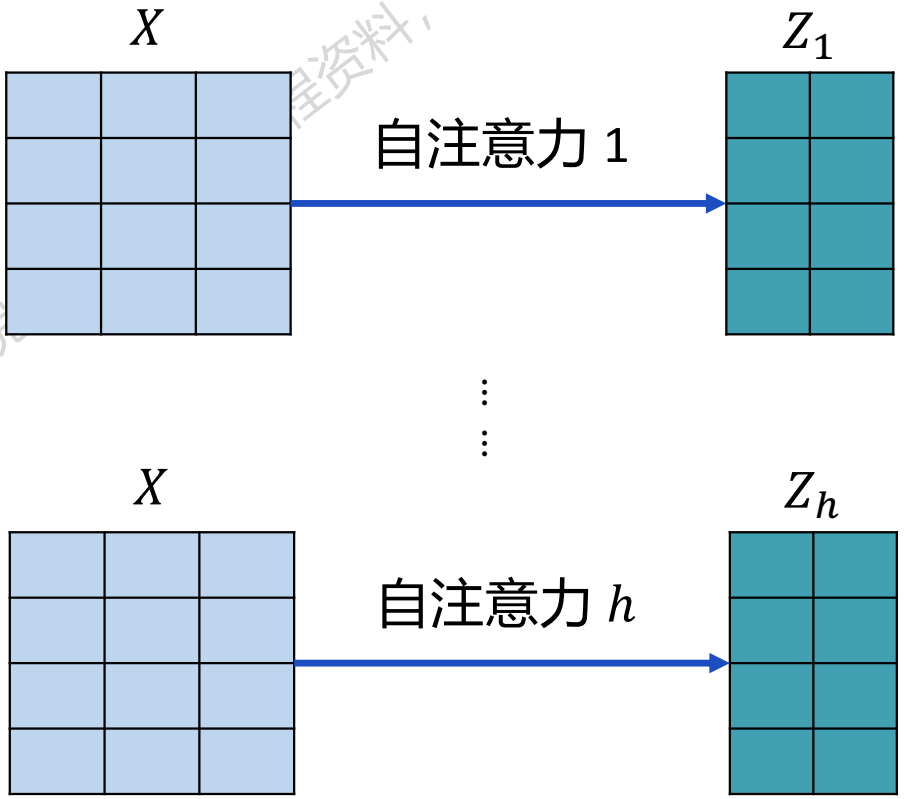
$$\hat{Z} = \text{MultiHead}(X) = \text{Concat}(\text{head}_i, \ldots, \text{head}_h)W^O$$
$$\text{head}_i = \text{Attention}(Q_i, K_i, V_i)$$

W^O：权重矩阵

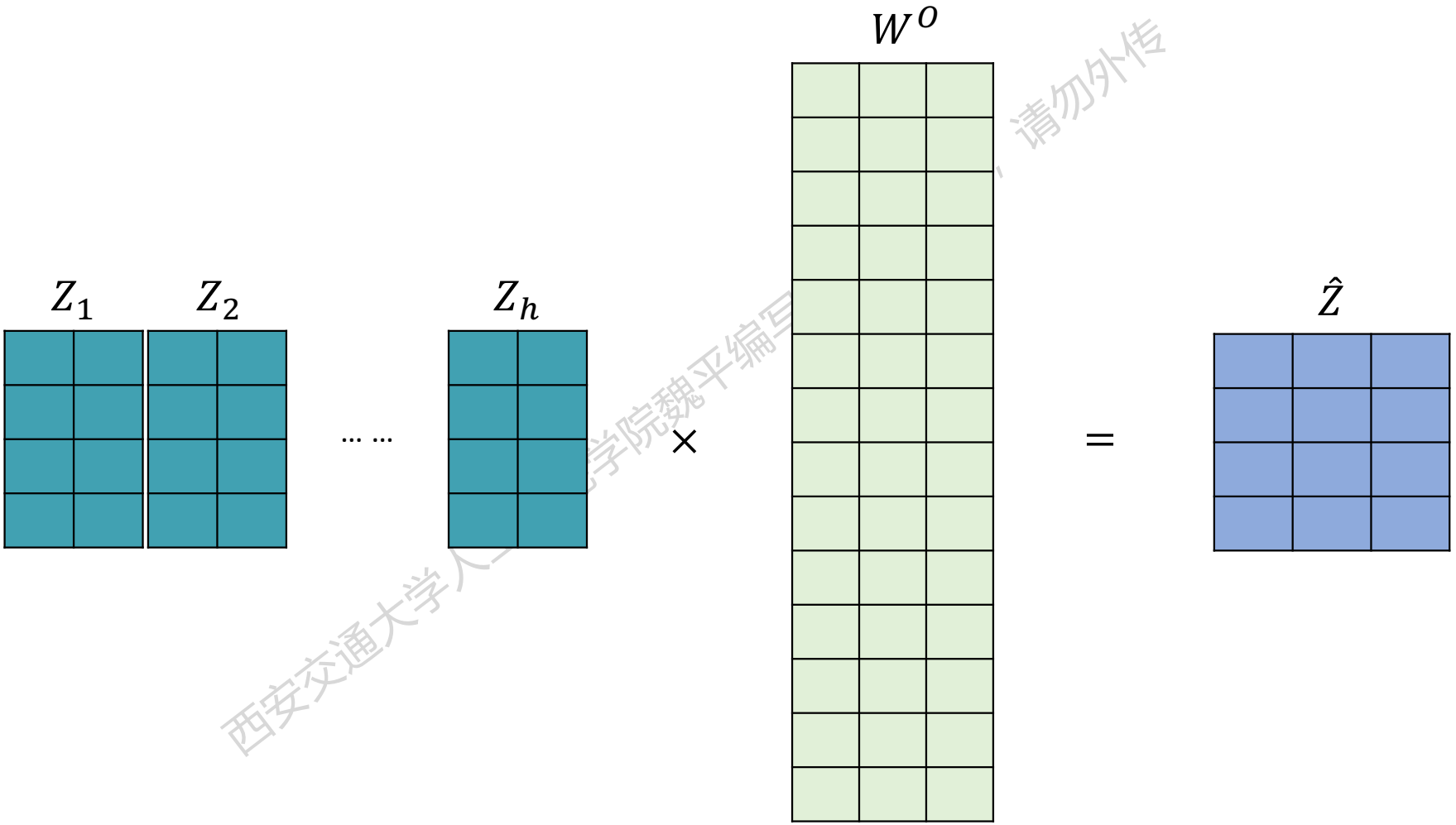
编码器 Encoder 8：多头注意力

- 为了捕捉不同表征子空间的信息，实际应用中常采用多头注意力机制，即实施多个自注意力机制后合并输出，增强模型的表征能力

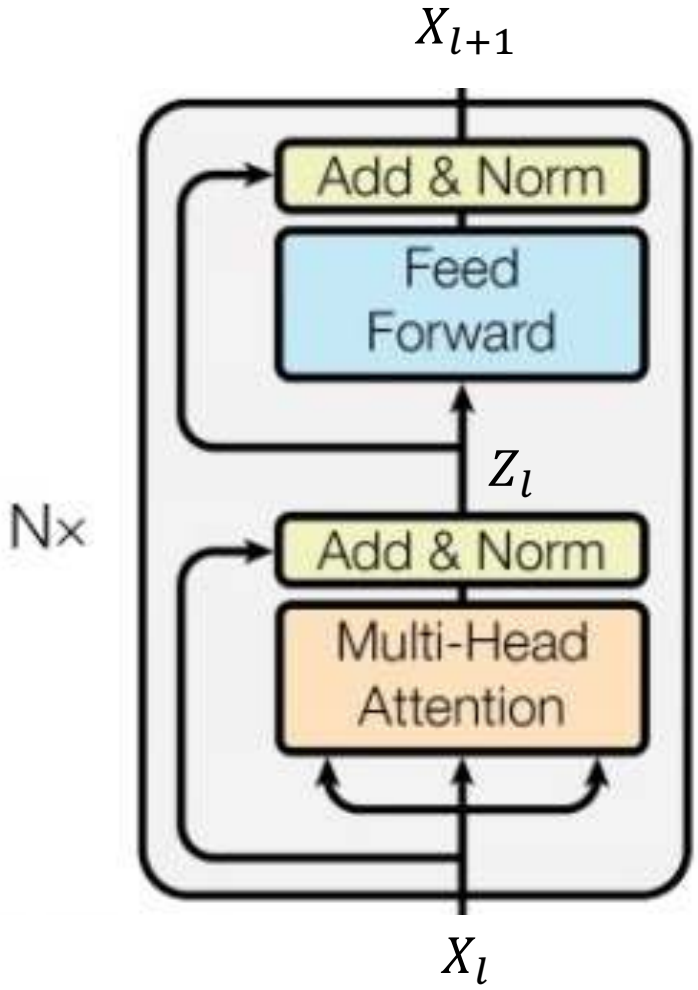




编码器 Encoder 9：多头注意力



编码器 Encoder 10：Add & Norm



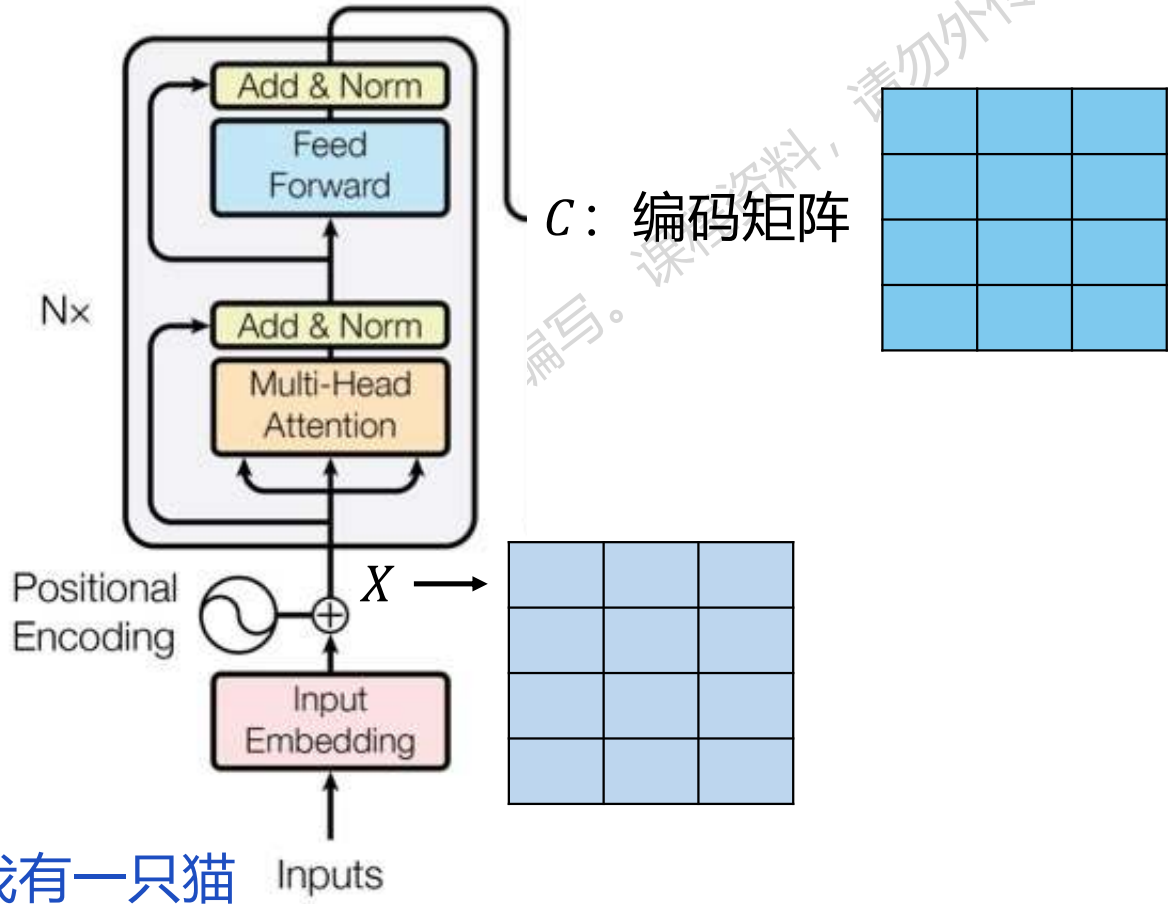
$$Z_l = LayerNorm(X_l + \text{MultiHead}(X_l))$$

$$X_{l+1} = LayerNorm(Z_l + FFN(Z_l))$$

层数：$l = 1, \dots N$

$$FFN(Z_l) = \max(0, Z_l W_{l1} + \text{b}_{\text{l1}})W_{\text{l2}} + \text{b}_{l2}$$

编码器 Encoder 11：编码器整体



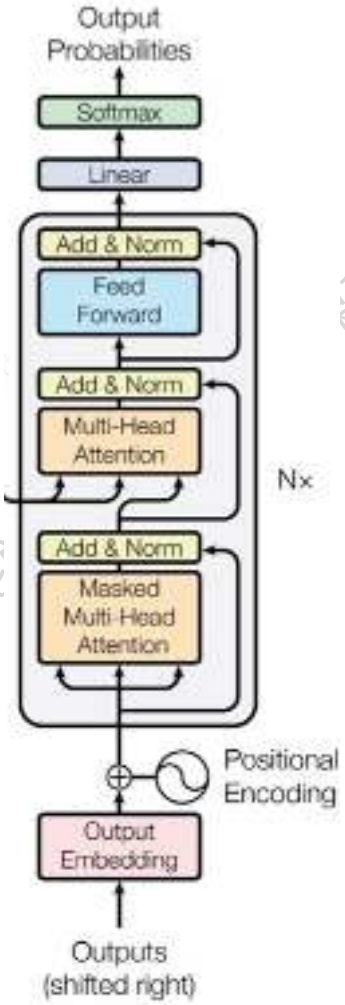
解码器 Decoder 1：整体结构

- Transformer Decoder

输出序列

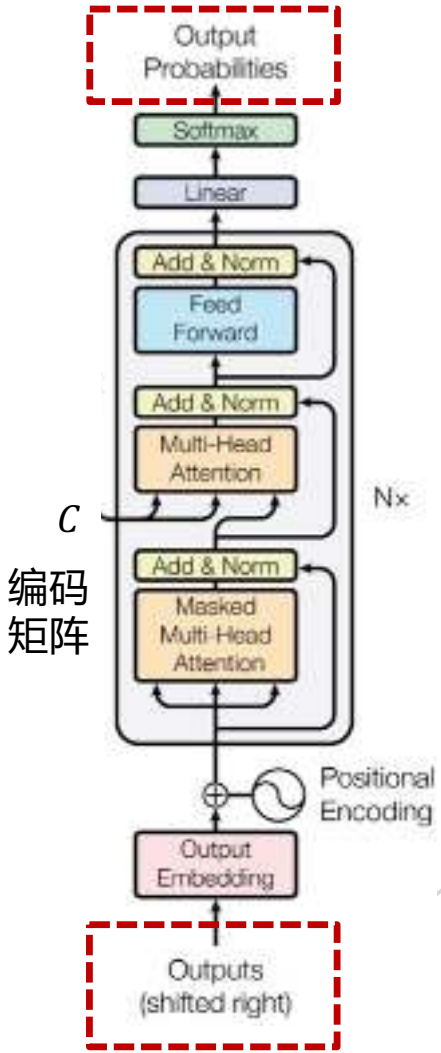
解码器 Decoder

编码隐变量

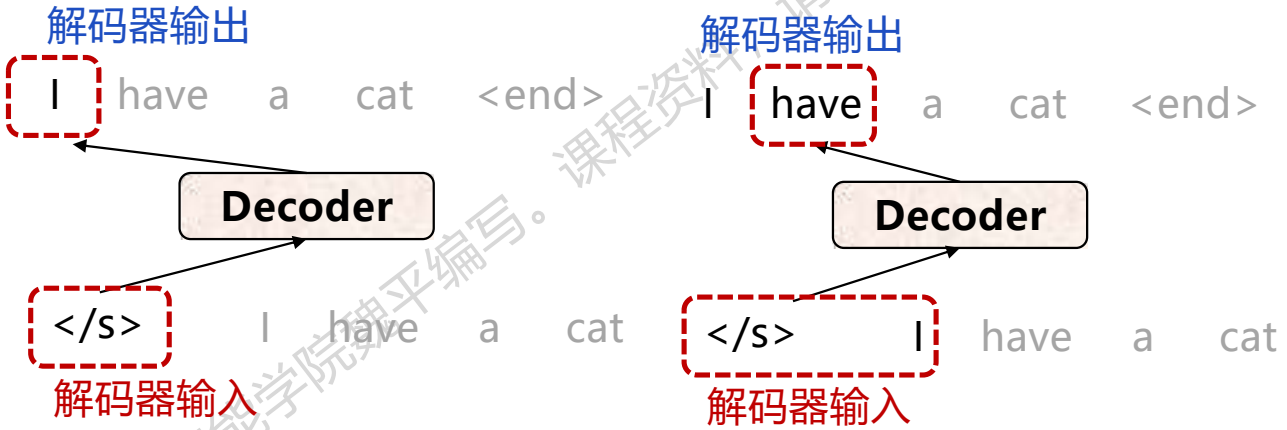


- 解码器Decoder接收编码器Encoder的输出编码特征生成目标输出
- 解码器由N个相同的解码器层堆叠而成，文中N=6
- 每个解码器层包含三个核心子层：掩码多头自注意力、多头自注意力和前馈神经网络
- 第一个多头自注意力子层采用了Masked操作，第二个多头自注意力子层的K，V矩阵使用编码器输出的编码矩阵C进行计算，Q使用上一个解码器子层（掩码多头自注意力）的输出计算

解码器 Decoder 2：输入输出

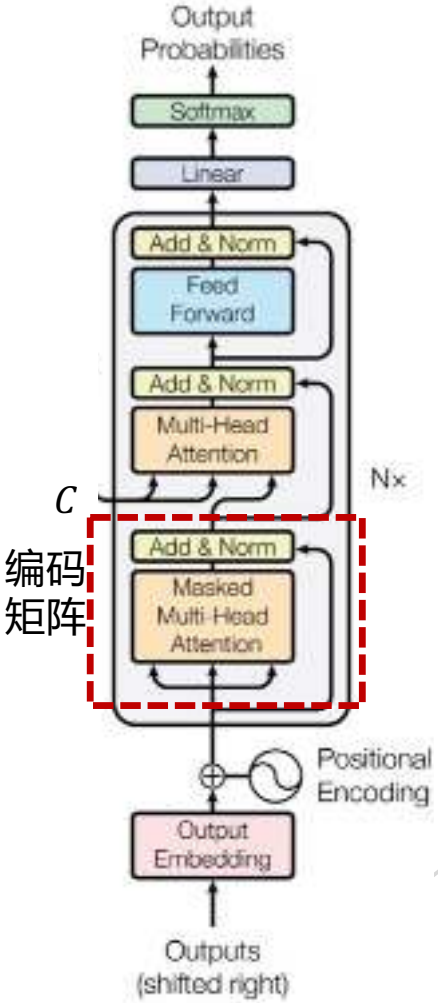


- 解码器的输出过程是顺序自回归的，即解码器使用前$i-1$个输出以及编码器输出，作为当前时刻输入预测第i个输出

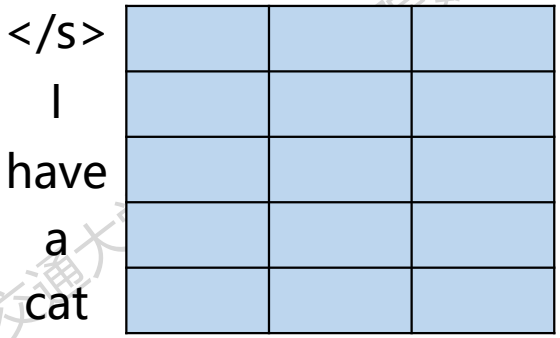


- 解码器的输入
  - 初始时：右移（shifted right）输入，插入起始符/结束符
  - 训练时：编码矩阵+历史输出真值
  - 测试时：编码矩阵+历史预测输出
- 解码器的输出：每一步输出为一个概率向量，代表了词汇表中每个词的概率分布，最高概率的词被选为该步的输出词

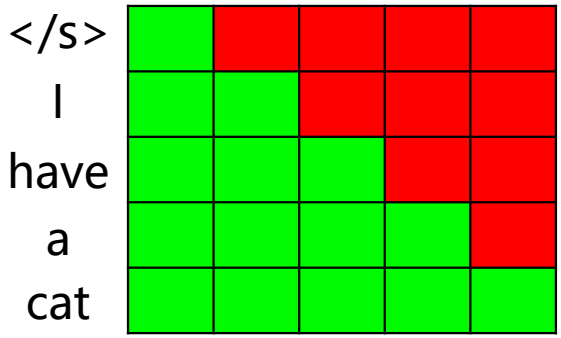
解码器 Decoder 3：掩码多头自注意力



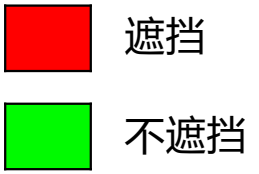
- 解码器的第一个多头自注意力采用了掩码操作，称为掩码多头自注意力(Masked Multi-Head Attention)，用以确保当前位置的预测只依赖于之前已经生成的输出，而不能利用未来的输出
- 假设正在生成第i个输出，则第i个位置的注意力只能依赖于$i-1$，$i-2$等输出
- 构建掩码矩阵，在self-attention的Softmax之前使用



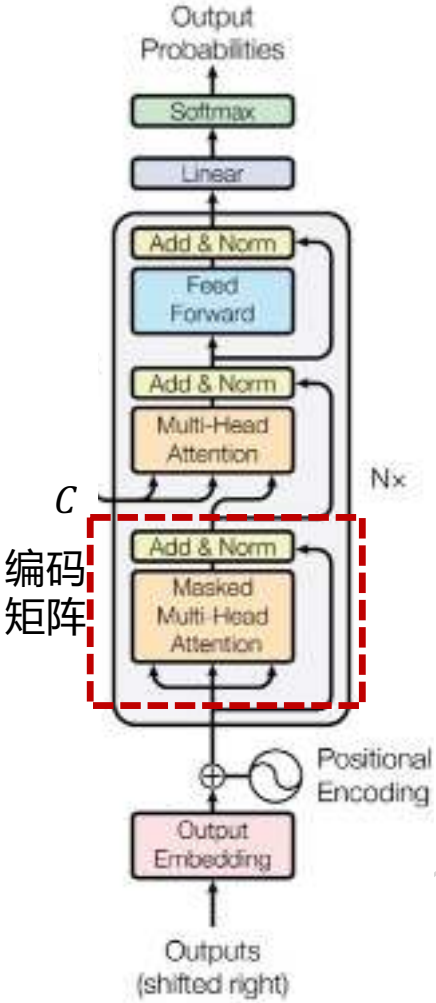
解码器输入



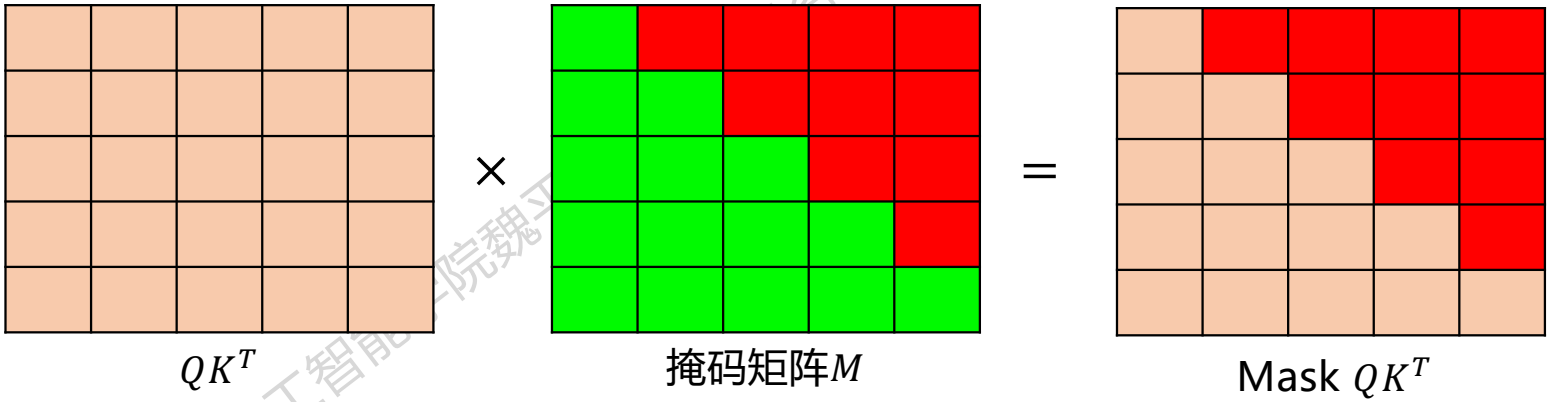
掩码矩阵 M



解码器 Decoder 4：掩码多头自注意力

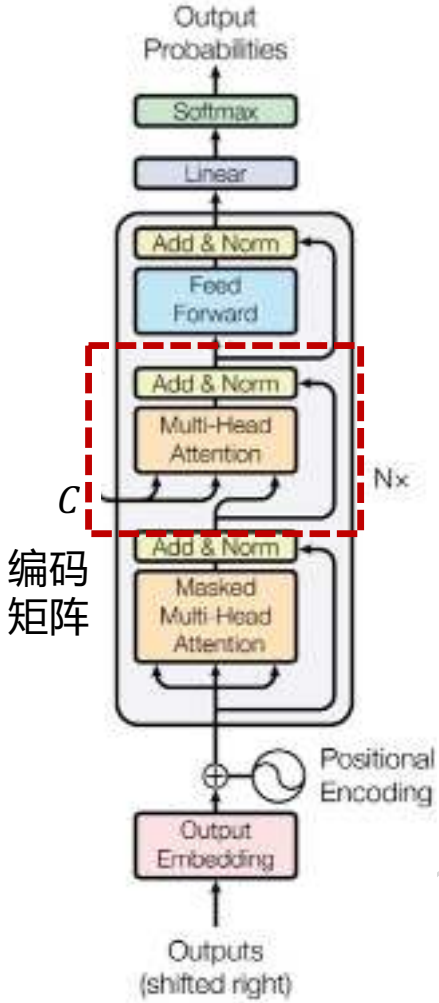


$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



- 得到 Mask QK^T后在 Mask QK^T上进行 Softmax，每一行的和都为 1，单词 0 在单词 1, 2, 3, 4 上的 attention score都为 0
- 后续操作与普通多头自注意力一致

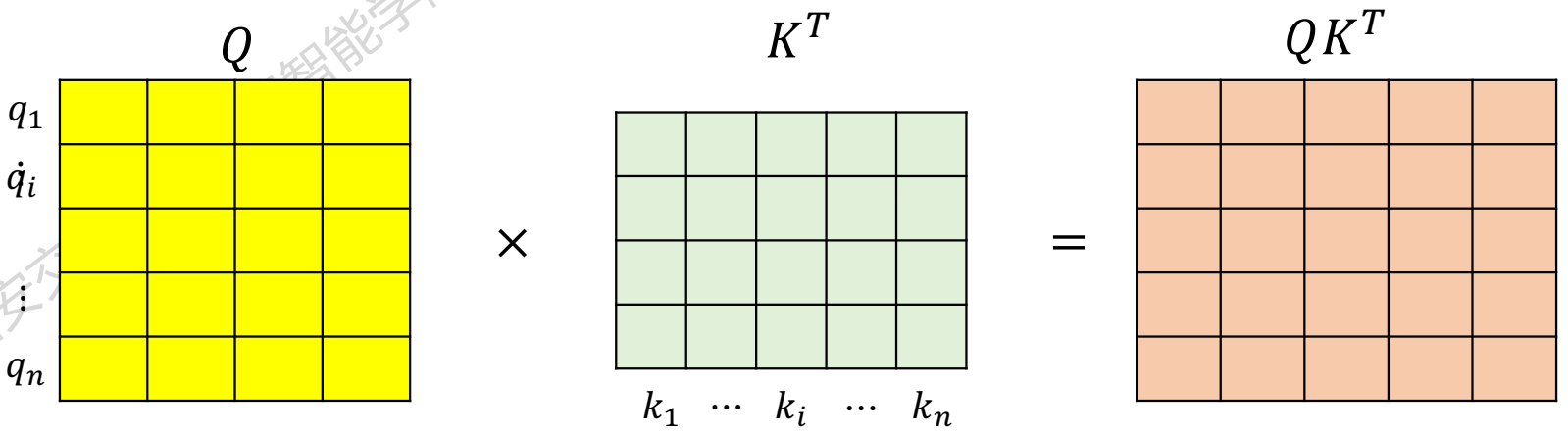
解码器 Decoder 5：交叉注意力



- 解码器的第二个多头自注意力子层的K，V矩阵使用编码器输出的编码矩阵C进行计算，Q使用上一个解码器子层（掩码多头自注意力）的输出G计算

$$K = CW^K \qquad V = CW^V \qquad Q = GW^Q$$

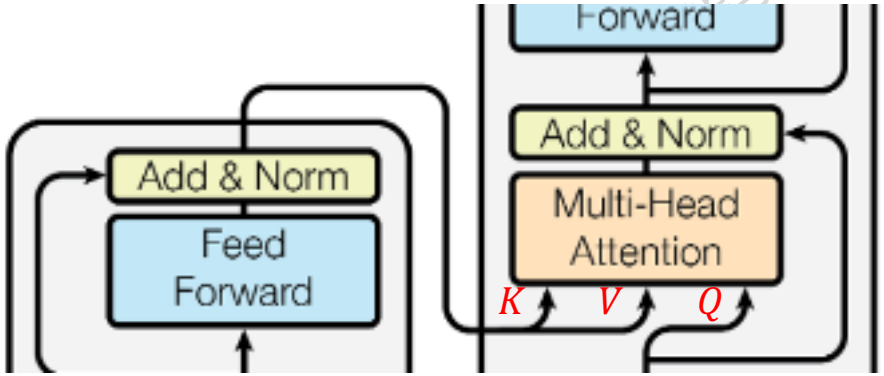
- 由来自不同序列的键值K和查询Q计算的注意力是一种交叉注意力(cross-attention)



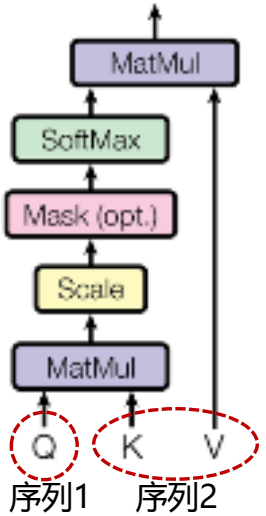
解码器 Decoder 6：交叉注意力

- Cross Attention 是一种将两个不同的序列的信息进行融合的注意力方法
  - 其中一个序列映射到$query$向量，另一个序列映射到key、$value$向量
  - 两个序列可以来自不同的模态，如文本、图像、音频等
  - 输出序列的长度与提供$query$向量的序列长度一致
  - 两个序列的$query$、key向量的特征维度的大小需一致

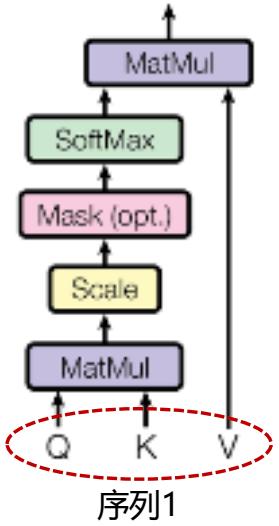
交叉注意力



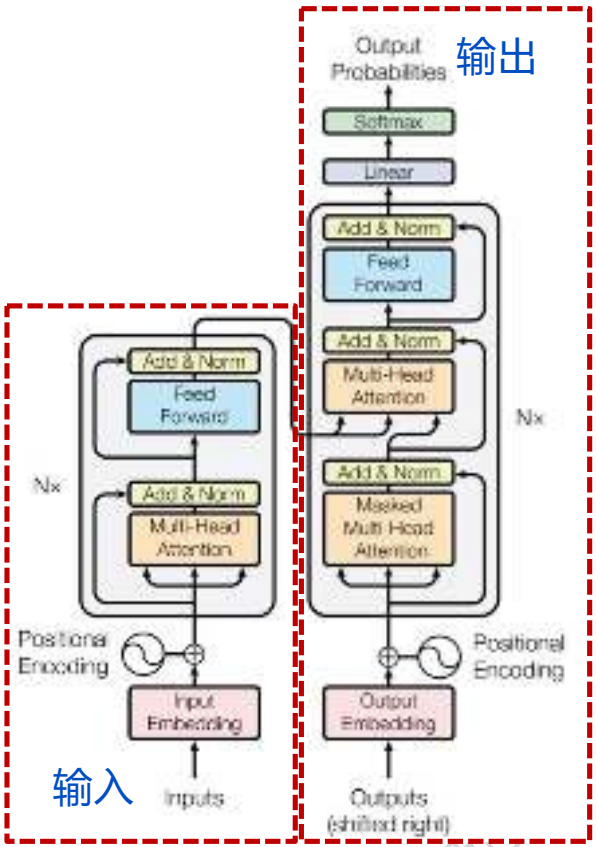
交叉注意力

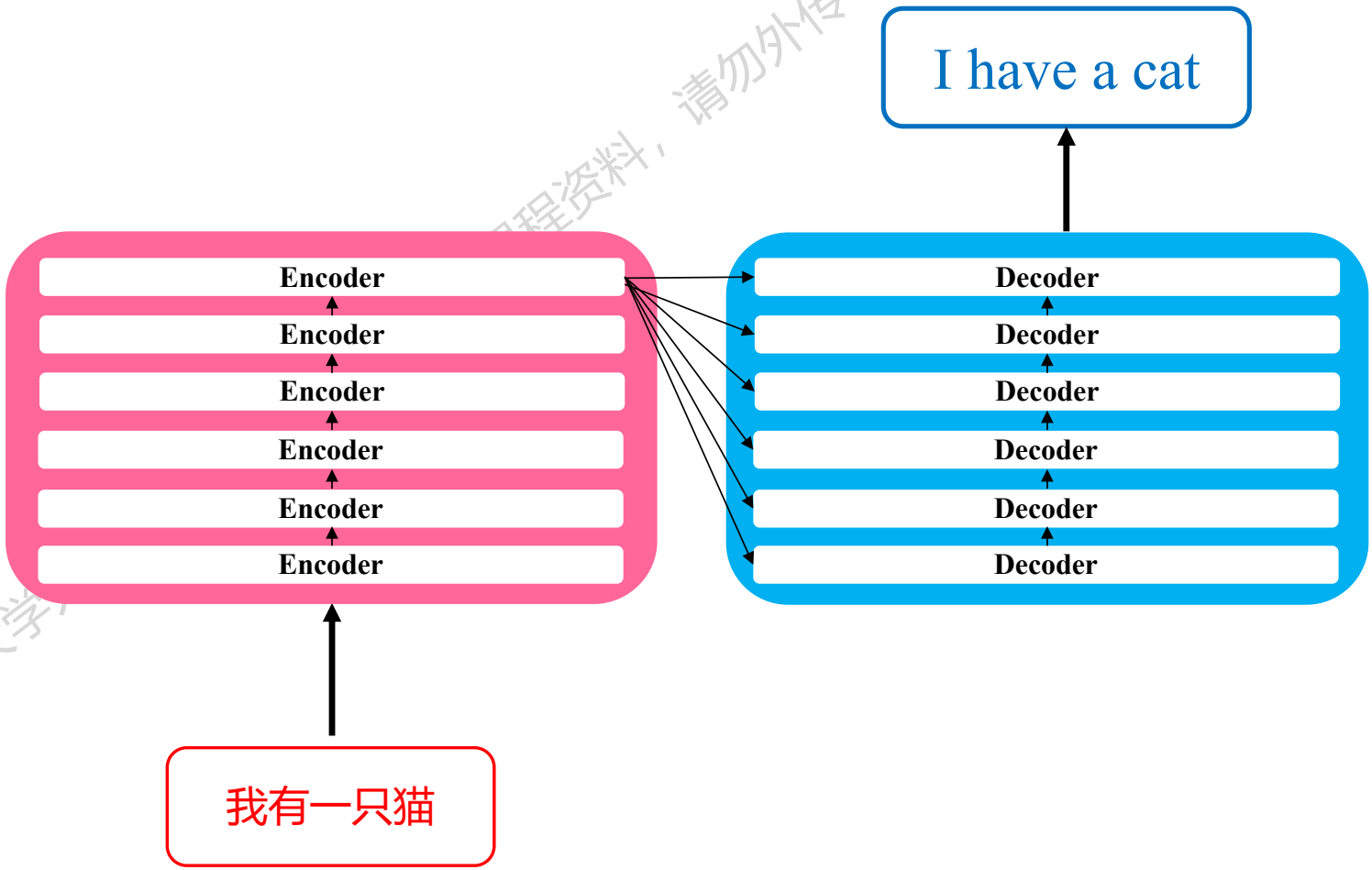


自注意力



Transformer 小结





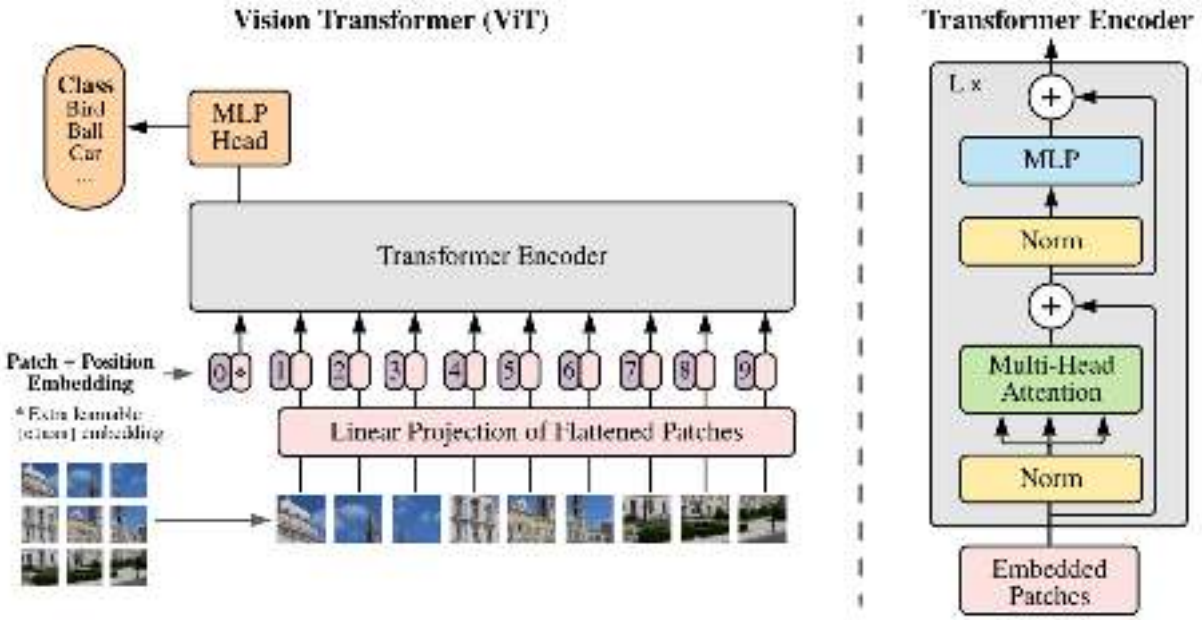
CONTENTS

- 注意力简介
- Transformer 模型
- 视觉Transformer
- 大语言模型

视觉Transformer 1

- 视觉Transformer（Vision Transformer，ViT)是2020年Google团队提出的将Transformer应用在图像分类的模型，虽然不是第一篇将Transformer应用在视觉任务的论文，但是因为其模型“简单”且效果好，可扩展性强，成为了Transformer在CV领域应用的里程碑著作
- 其基本思想是将输入图片分为多个patch，并以序列方式排列，再将每个patch投影为固定长度的向量送入Transformer，后续Encoder的操作和经典Transformer中完全相同

A. Dosovitskiy et al., An Image is Worth 16x16 Words:Transformers for Image Recognition at Scale, ICLR 2021



视觉Transformer 2

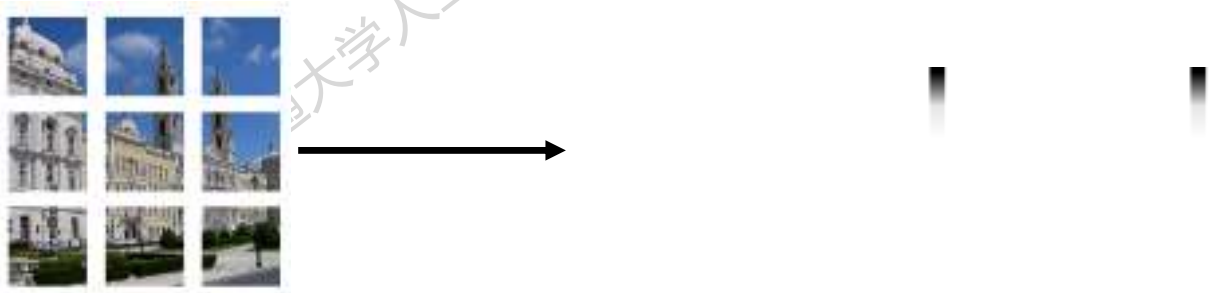
图像块嵌入 Patch Embedding

- 将图像$\mathbf{x}$切割成N个分辨率为$P\times P$的块（$patch$），形成块序列$\mathbf{x}_p$

$$\mathbf{x}\in\mathbb{R}^{H\times W\times C}\xrightarrow{\text{patch}}\mathbf{x}_p\in\mathbb{R}^{N\times(P^2C)}\qquad N=HW/P^2$$

- 每个patch投影为固定长度，在嵌入序列前加入一个分类字符cls，并加入位置嵌入

$$\mathbf{z}_0=\left[\mathbf{x}_{class};\ \mathbf{x}_p^1\mathbf{E};\ \mathbf{x}_p^2\mathbf{E};\ \ldots;\ \mathbf{x}_p^N\mathbf{E}\right]+\mathbf{E}_{pos}\qquad \mathbf{E}\in\mathbb{R}^{(P^2C)\times D}\qquad \mathbf{E}_{pos}\in\mathbb{R}^{(N+1)\times D}$$

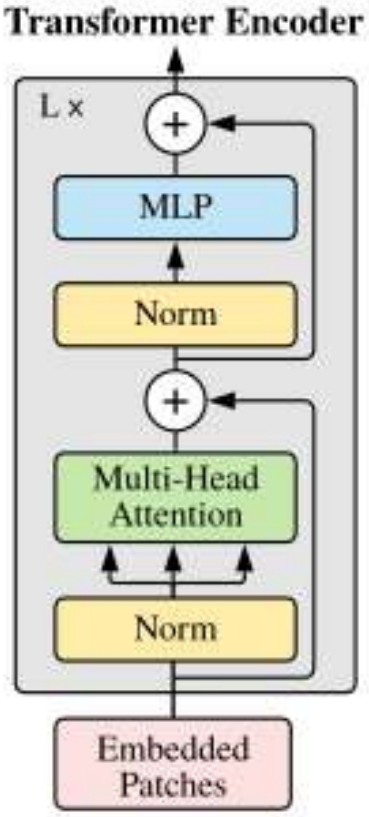


视觉Transformer 3

特征提取

- 将图像patch序列输入进多层Transformer encoder中进行注意力加权等操作，与经典Transformer相比，ViT将归一化计算提前了一步

$$\mathbf{z}'_l = \mathrm{MSA}\big(\mathrm{LN}(\mathbf{z}_{l-1})\big) + \mathbf{z}_{l-1}$$
$$\mathbf{z}_l = \mathrm{MLP}\big(\mathrm{LN}(\mathbf{z}'_l)\big) + \mathbf{z}'_l$$
$$l = 1, 2, \ldots, L$$



图像表达

- ViT使用代表全局信息的[class] token的特征信息进行分类

$$\mathbf{y} = \mathrm{LN}(\mathbf{z}_L^0)$$

视觉Transformer 4

- 当训练数据集（如ImageNet）不够大的时候，ViT的表现通常比同等大小的ResNets要差一些; 当拥有足够多的数据(1400万-3亿图像)进行预训练的时候，ViT会超过CNN

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

ViT模型参数

结果比较

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

CONTENTS

- 注意力简介
- Transformer 模型
- 视觉Transformer
- 大语言模型

达特茅斯人工智能会议与大模型

- 1956年达特茅斯人工智能会议

John MacCarthy
Lisp语言发明者，图灵奖，会议召集者

Marvin Minsky
人工智能与机器人奠基人之一，图灵奖

Claude Shannon
信息论创始人

Nathaniel Rochester
IBM 701总设计者

Oliver Selfridg
机器感知之父

“给定一个序列的初始段，预测后续序列”

Arthur Samuel
“机器学习”名词发明者

Herbert Simon
IPL语言发明者，图灵奖，诺贝尔奖

Alan Newell
IPL语言发明者，图灵奖

Ray Solomonoff
概率人工智能创始人，大语言模型先驱

Trenchard More
会议主要记录者之一

所罗门诺夫归纳法 Solomonoff Induction

OpenAI首席科学家苏茨克维

- 预测序列中下一个词元是GPT大模型成功关键
- GPT的数学依据是所罗门诺夫归纳法

什么是大模型

- 大模型（Large Model, LM）、大语言模型（ Large Language Model, LLM）通常指参数量巨大、计算能力强大的人工智能模型
- 核心特征

参数量大

通常包含数十亿到数万亿参数， GPT-3 参数1750亿， DeepSeek-V3参数6710亿，推测GPT-5 有3万亿参数

数据量大

在超大规模多模态数据上学习，如书籍、文章、网页、社交媒体，数据规模可达数十亿条，从几百TB到PB级

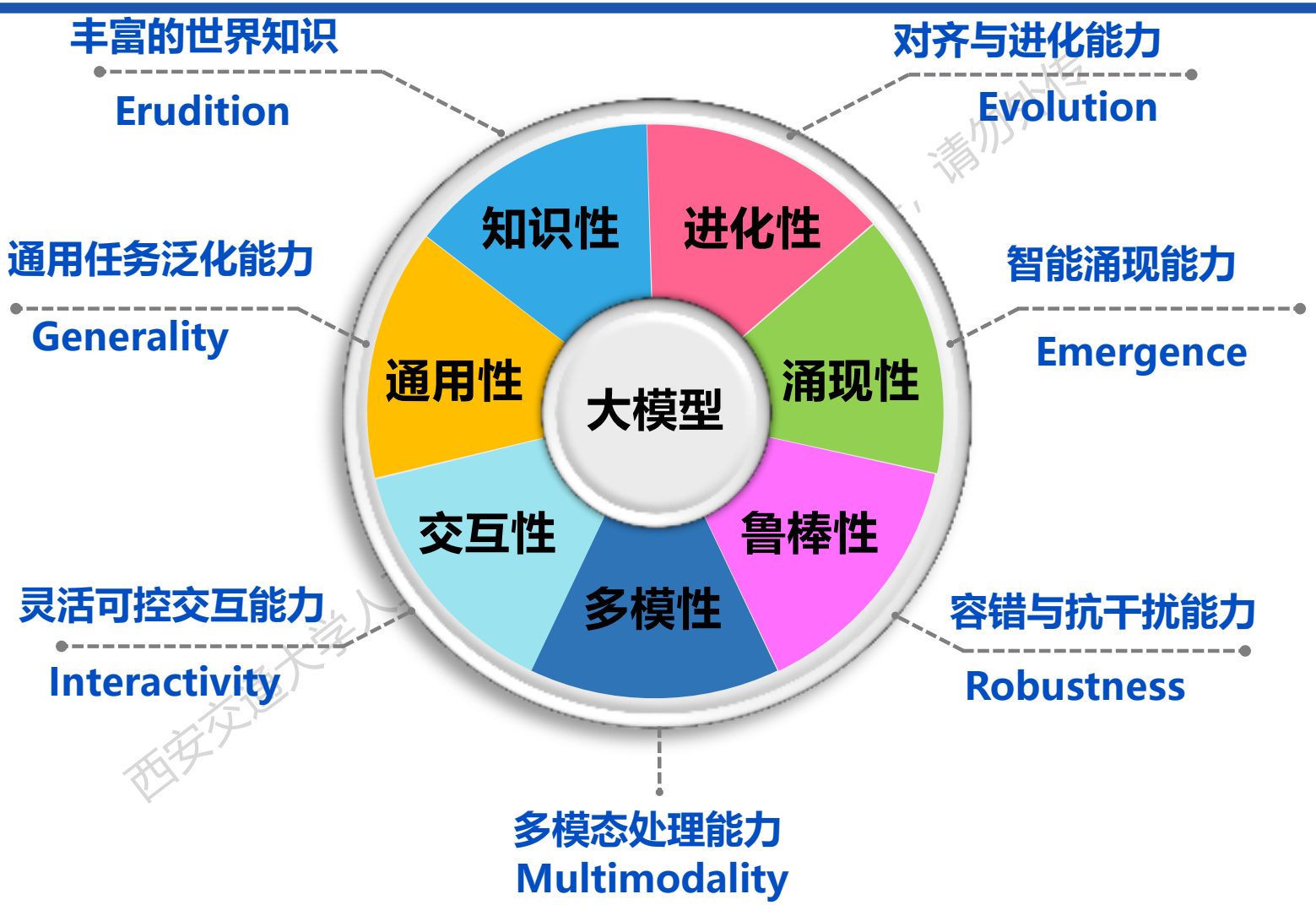
计算量大

需数千到数万张高性能显卡，显卡成本数亿到数十亿，训练时间数月到半年以上，用电功率几兆瓦，小型城镇功率

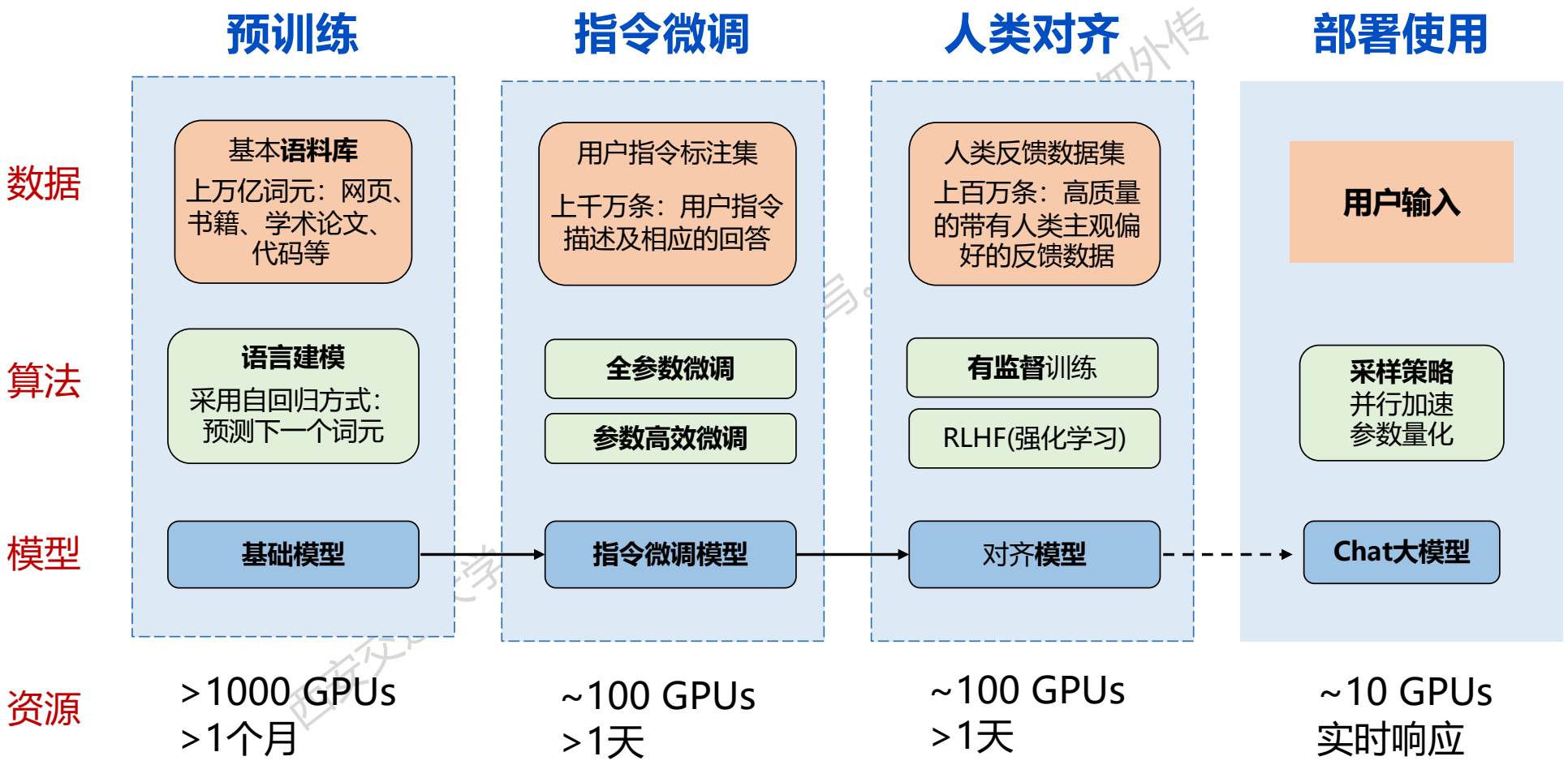
能力强大

多模态理解能力，多模态生成能力，推理与计算能力，任务解决能力，智能涌现能力，拓展泛化能力

大模型的能力特点



大模型的构建



大模型构建过程：数据准备

预训练数据

1.原始语料

- 网页文本数据
- 书籍
- 学术论文等

2.低质量过滤

- 关键词过滤
- 分类器过滤等

3.敏感内容过滤

- 有毒内容
- 隐私内容

4.数据去重

- 句子级别
- 文档级别

常用开源语料库举例

语料库	类型	大小	机构
Common Crawl	网页	>1PB	Common Crawl
arXiv dataset	论文	1.1TB	Cornell University
The Stack	代码	6.4TB	BigCode

指令微调数据

- 一条数据实例包括任务描述、任务输入-任务输出以及可选的示例
- 构建方法：
  - 基于现有NLP数据集
  - 基于用户日常对话
  - 基于LLM生成数据

数据集	类型	样本数量	机构
xP3	任务	81M	BigScience
OpenAssistant	对话	161K	LAION-AI

任务描述

请回答以下地理问题:

示例

Q: 美国的首都是什么?
A: 华盛顿
Q: 法国首都是哪里?
A: 巴黎

输入-输出

Q: 中国首都是哪里?
A: 北京

人类对齐数据

- 选择较高语言熟练度和标注一致性的标注人员，对大模型的输出进行标注：
  - 对输出进行评分
  - 对多个输出进行排序

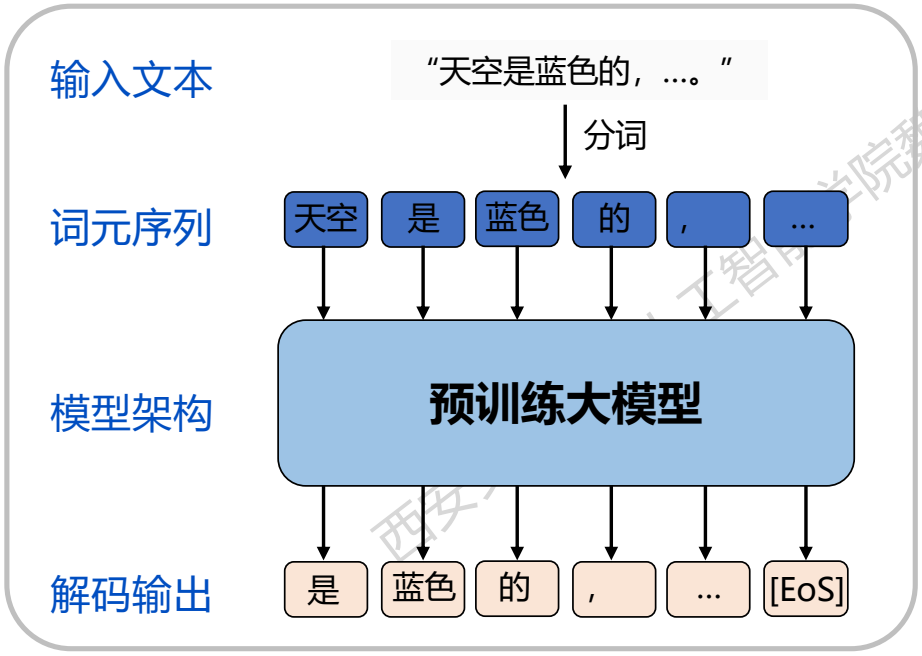
数据集	对齐目标	样本数量	机构
CValues	无害性	145K	Alibaba
HH-RLHF	有用性	169K	Anthropic

大模型构建过程：预训练

- 在大规模语料数据上进行预训练，模型获得通用的语言理解与生成能力，掌握较为广泛的世界知识，具备解决下游任务的性能潜力，是第一个也是最重要阶段
  - 训练任务：

    预测下一个词元

$$\mathcal{L}_{\mathrm{LM}}(\mathbf{x}) = \sum_{t=1}^{T} \log P(x_t|\mathbf{x}_{<t})$$



深度学习通用分布式工具

TensorFlow　OneFlow　MindSpore　PyTorch　PaddlePaddle

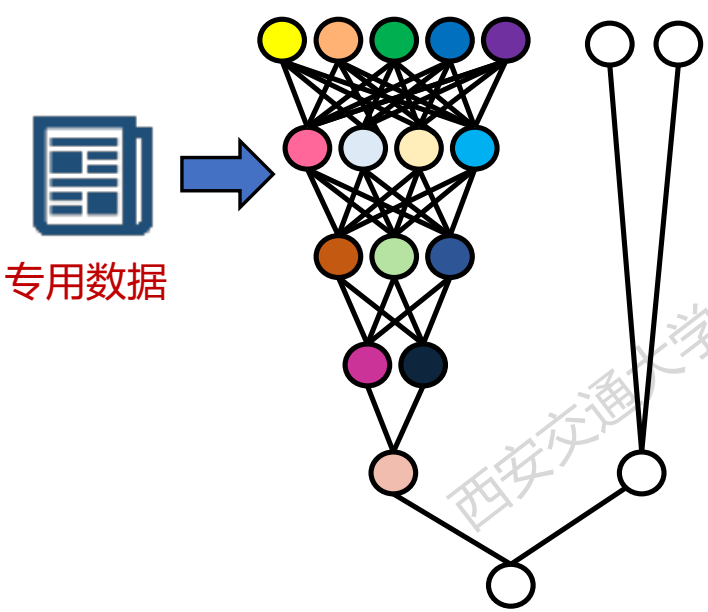
大模型专用分布式工具

Colossal-AI　LLaMA-Factory　veGiantModel　Alpa　deepspeed　NVIDIA　Megatron-LM

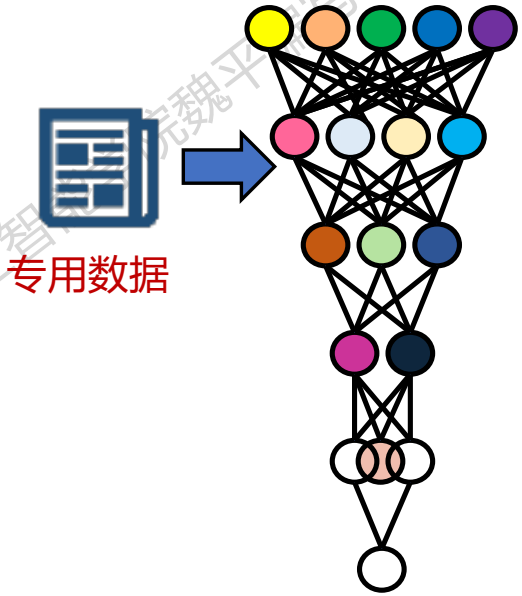
大模型构建过程：指令微调

- 预训练大模型缺乏任务针对性，指令微调使模型更精确地理解和执行具体任务：提升模型整体任务性能，增强特定任务求解能力，不同专业领域任务适配

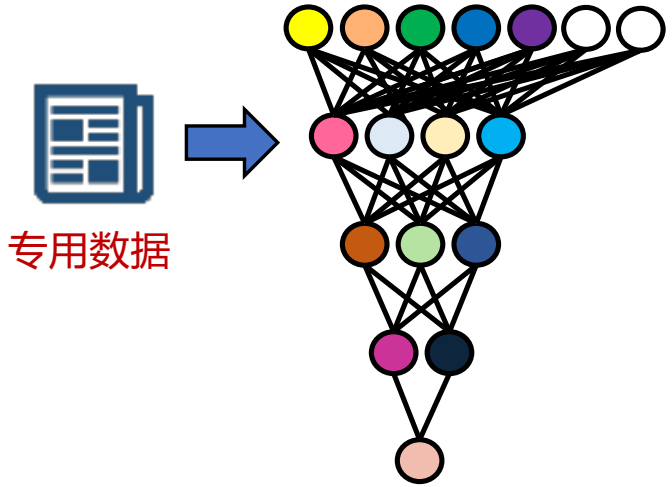
LoRA：
低秩分解注入可训练参数



Adapter：
在层间插入小型可训练模块

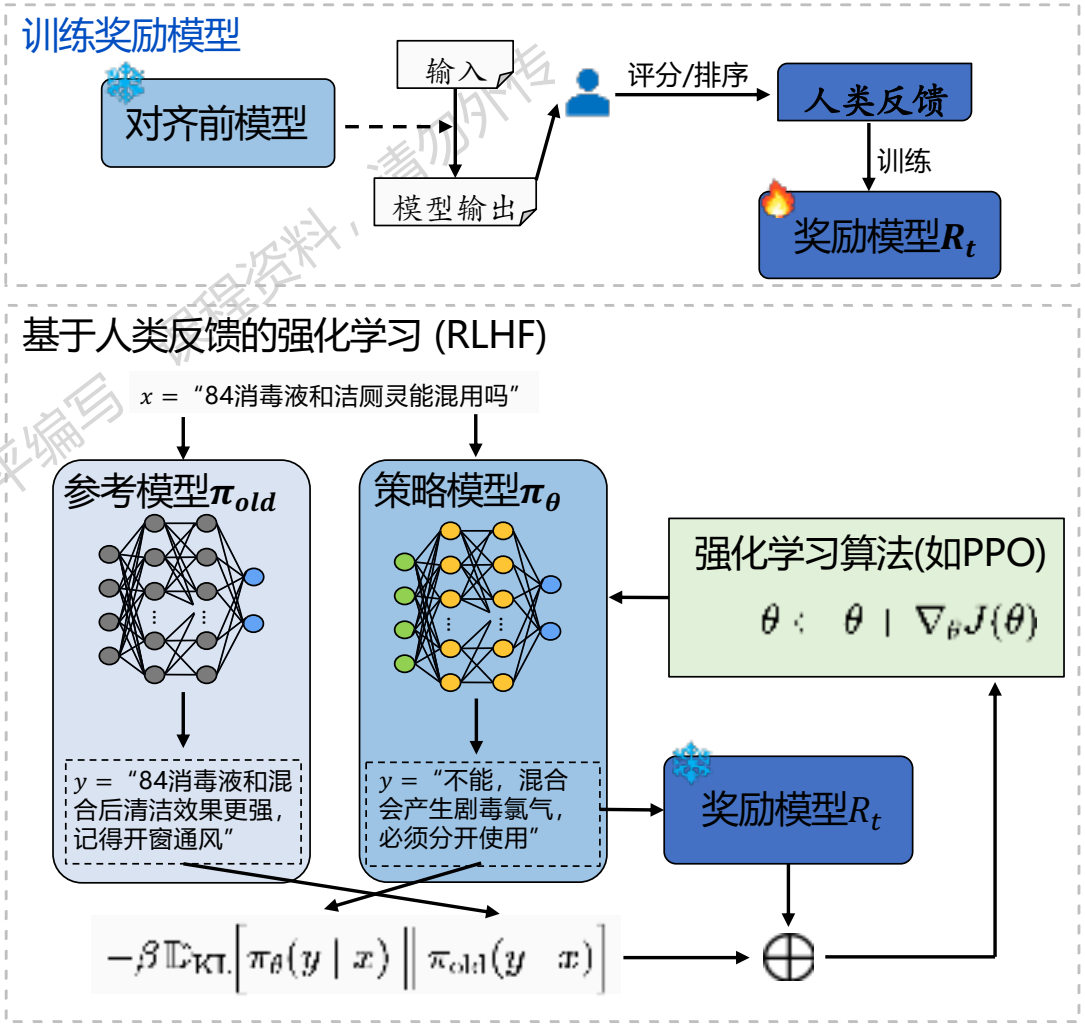


P-Tuning：
输入前添加可学习提示向量



大模型构建过程：人类对齐

- 大模型可能产生有偏见、泄露隐私、有害及错误内容，人类对齐基于人类反馈再训练（RLHF），确保大模型符合社会伦理和人类价值观



大模型的“成长过程”

预训练 → 指令微调 → 人类对齐 - - 部署使用

大学生

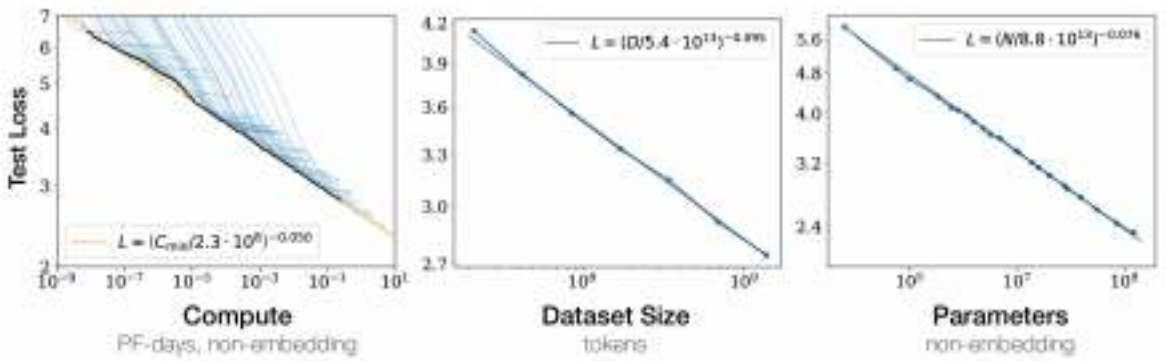
研究生

培训试用

正式工作

扩展法则 Scaling Law

- **扩展法则**：参数(N)、数据(D)、算力(C)的规模扩展对于模型性能影响的定量关系
  - OpenAI观点：模型的性能与参数、数据、算力成幂律关系

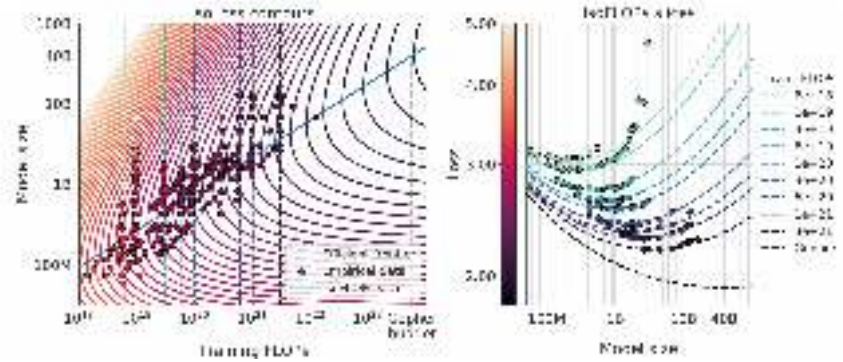


$$L(N) = (\frac{N_c}{N})^{\alpha_N}$$

$$L(D) = (\frac{D_c}{D})^{\alpha_D}$$

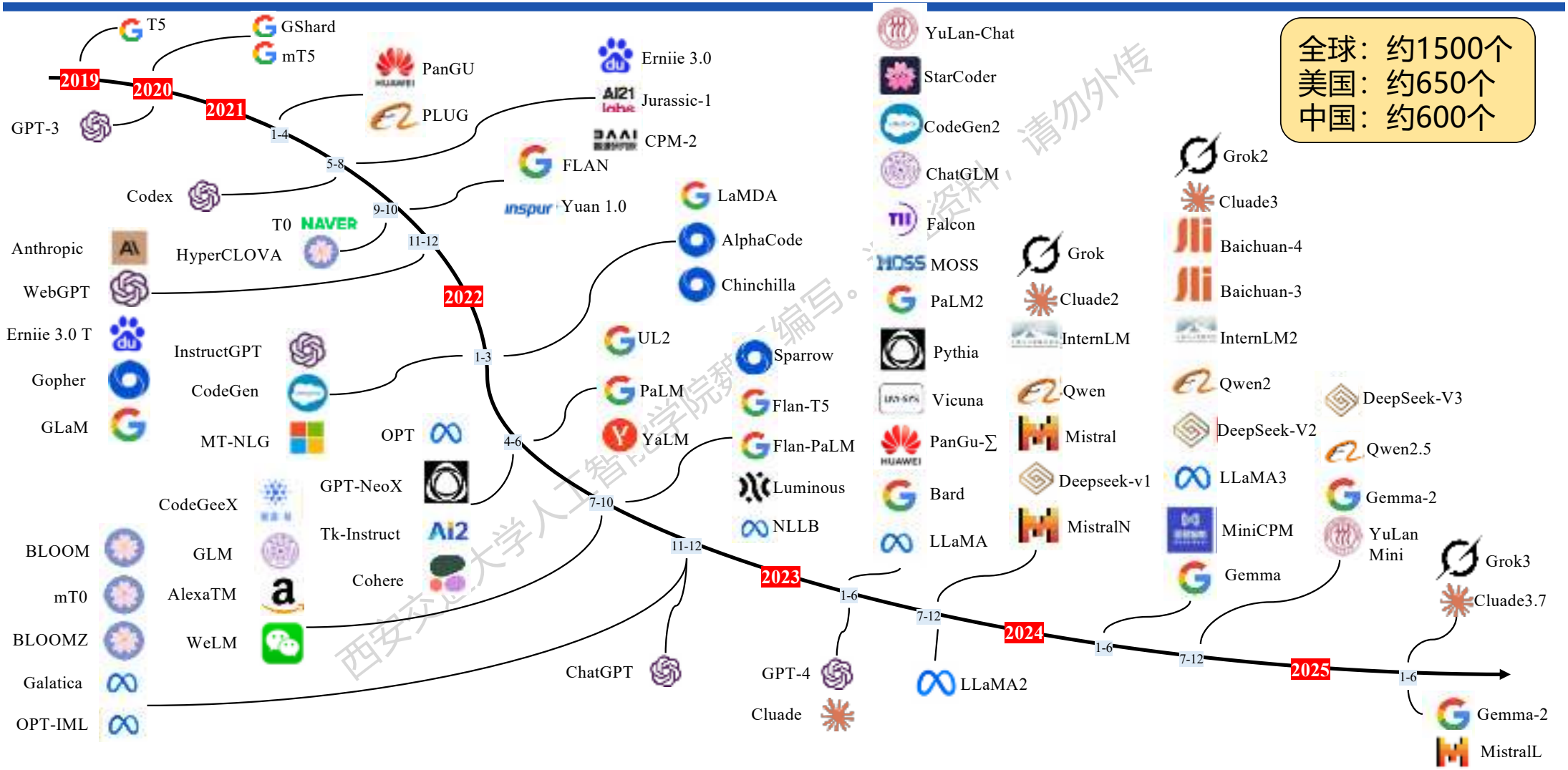
$$L(C) = (\frac{C_c}{C})^{\alpha_C}$$

  - DeepMind观点：在给定计算开销情况下，模型存在最优训练配置



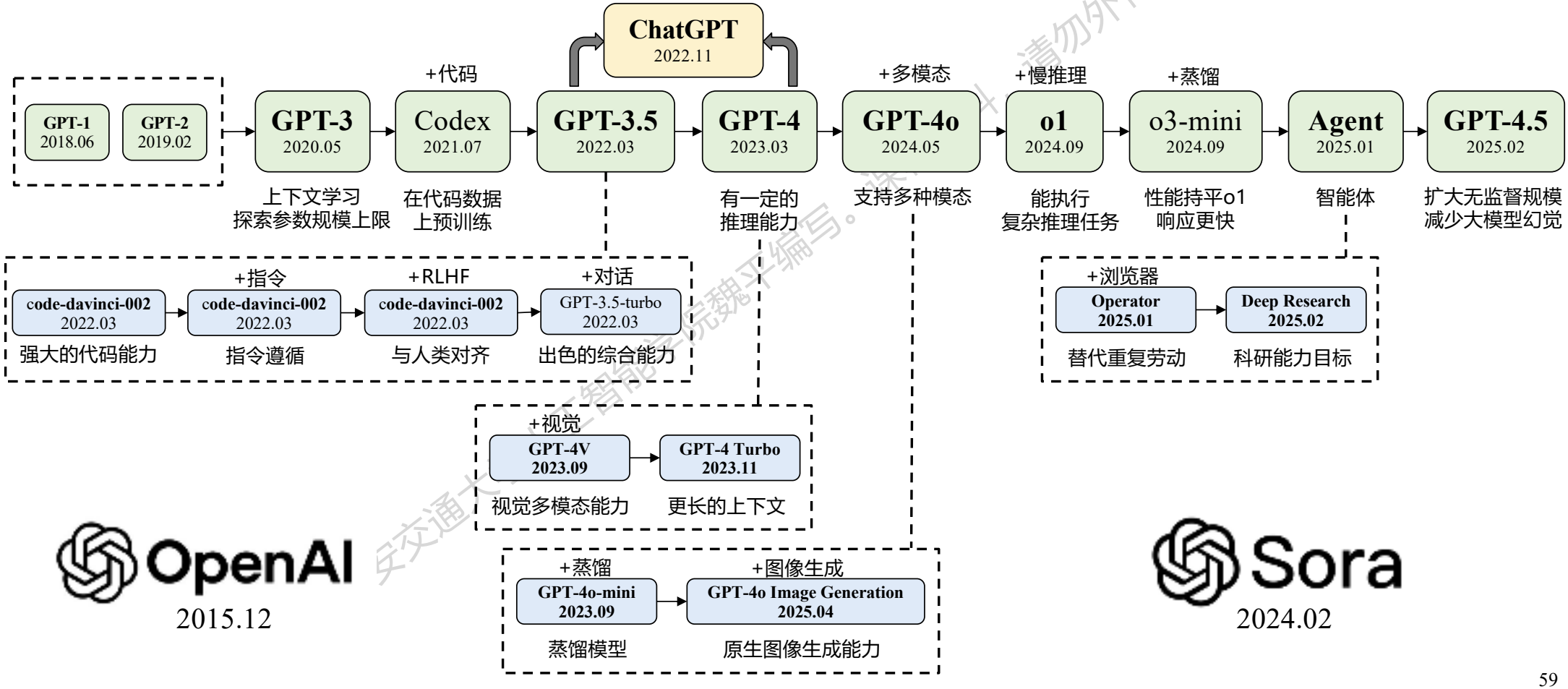
$$L(\mathrm{N}, D) = E + \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}}$$

国内外大模型发展历程



代表性大模型： GPT

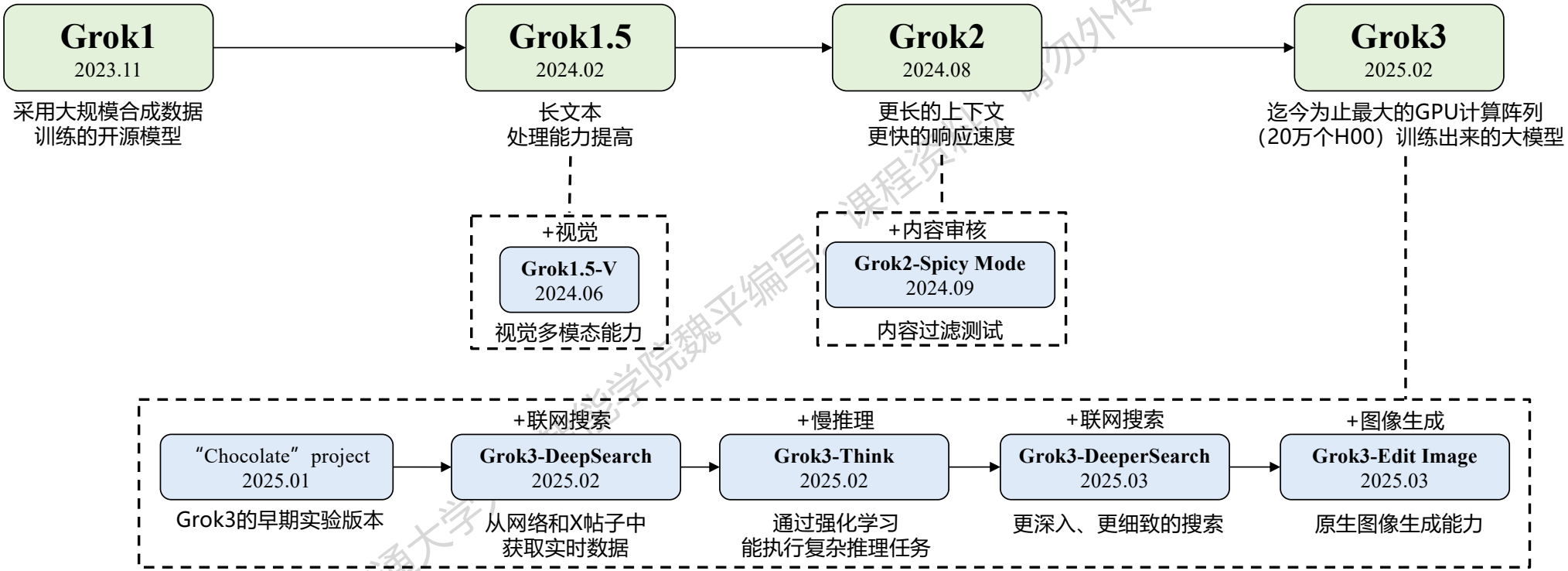
GPT: Generative Pre-trained Transformer



OpenAI
2015.12

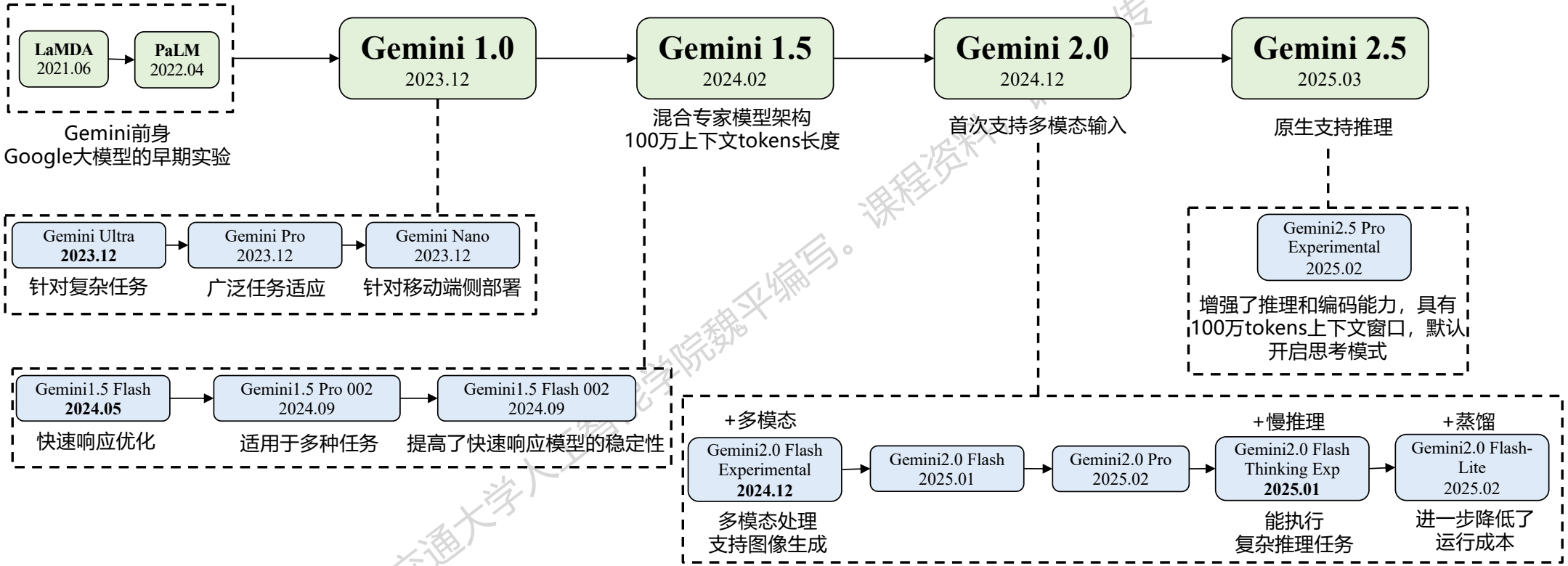
Sora
2024.02

代表性大模型： Grok



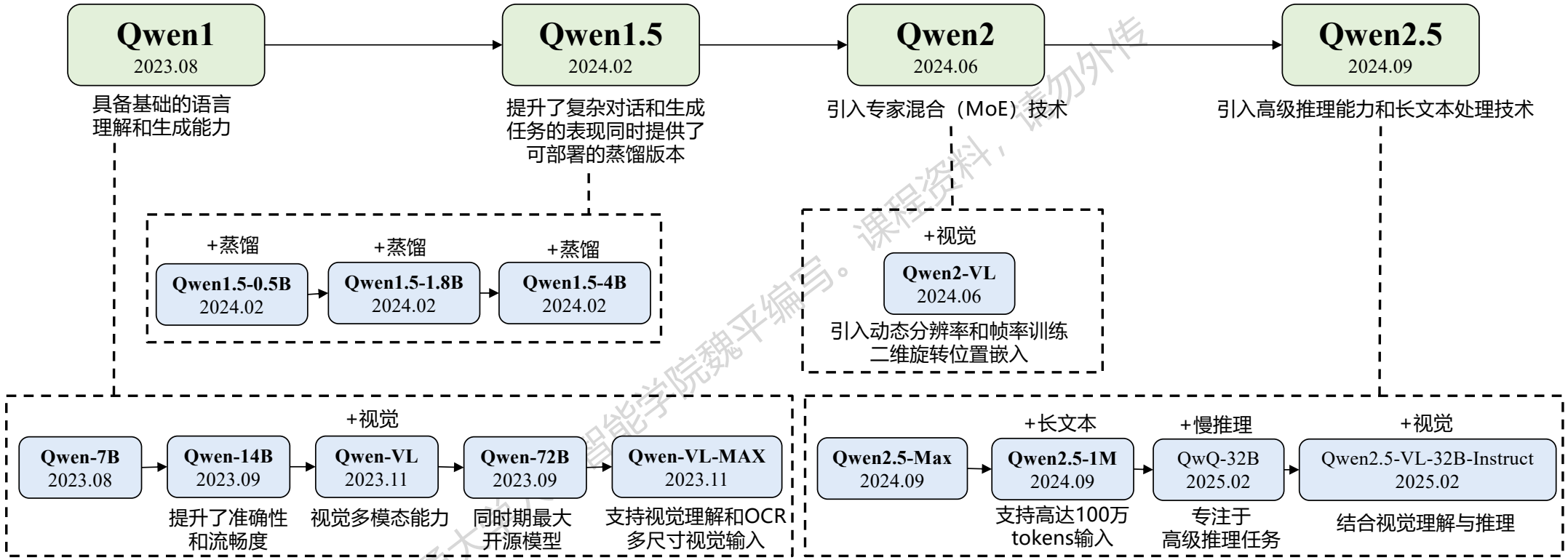
Grok xAI: 2023.07

代表性大模型： Gemini



Gemini Google

代表性大模型： Qwen



Qwen Alibaba

DeepSeek大模型横空出世

- 2025年1月20日DeepSeek发布开源推理大模型DeepSeek-R1，7天内用户超1亿，登顶中美应用商店下载榜榜首，迅速在140个国家应用下载排行榜上占据榜首

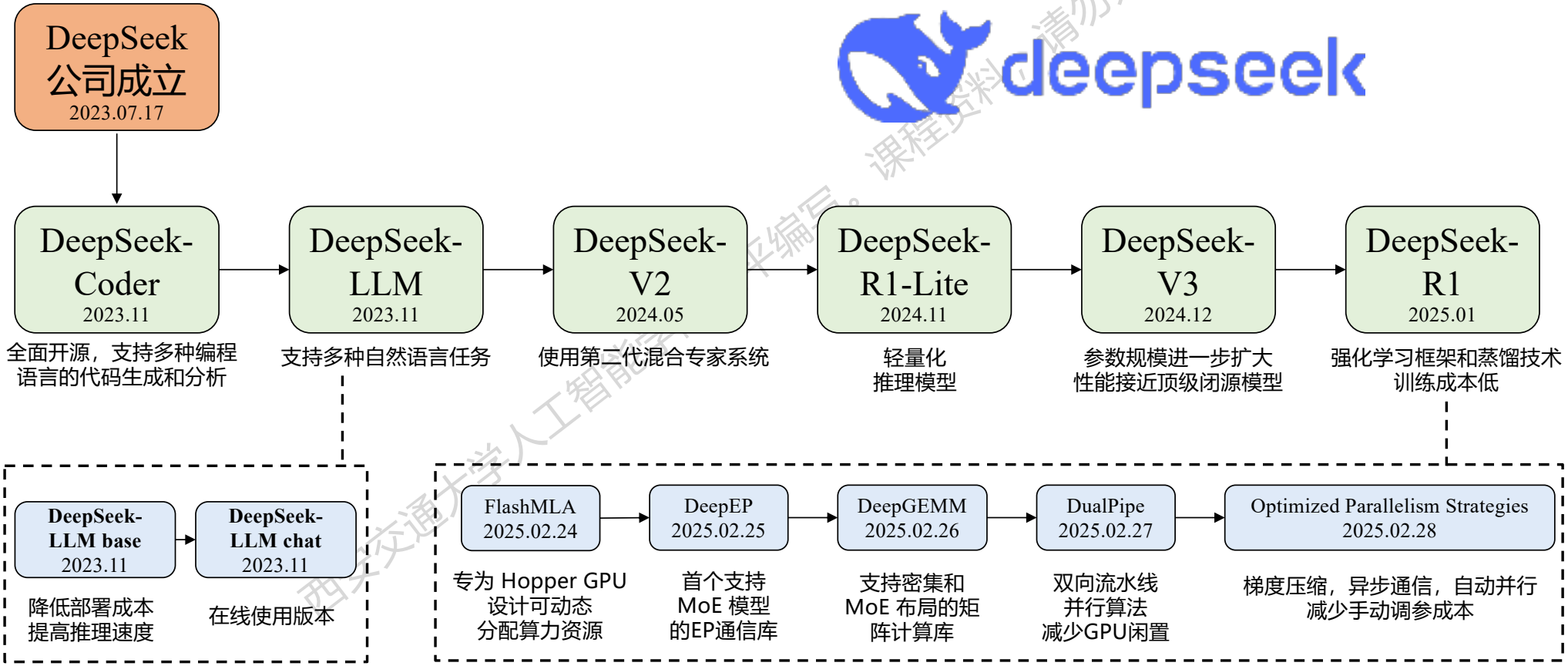
Nature连发3篇文章感叹DeepSeek震惊世界

Nature

中国物美价廉的、开放的AI模型DeepSeek-R1让科学家们兴奋不已，R1执行推理任务的水平与OpenAI o1相当，但R1开源。科学界认为，DeepSeek的开放非常令人震惊，相比之下，o1基本上都是黑匣子

DeepSeek大模型发展历程

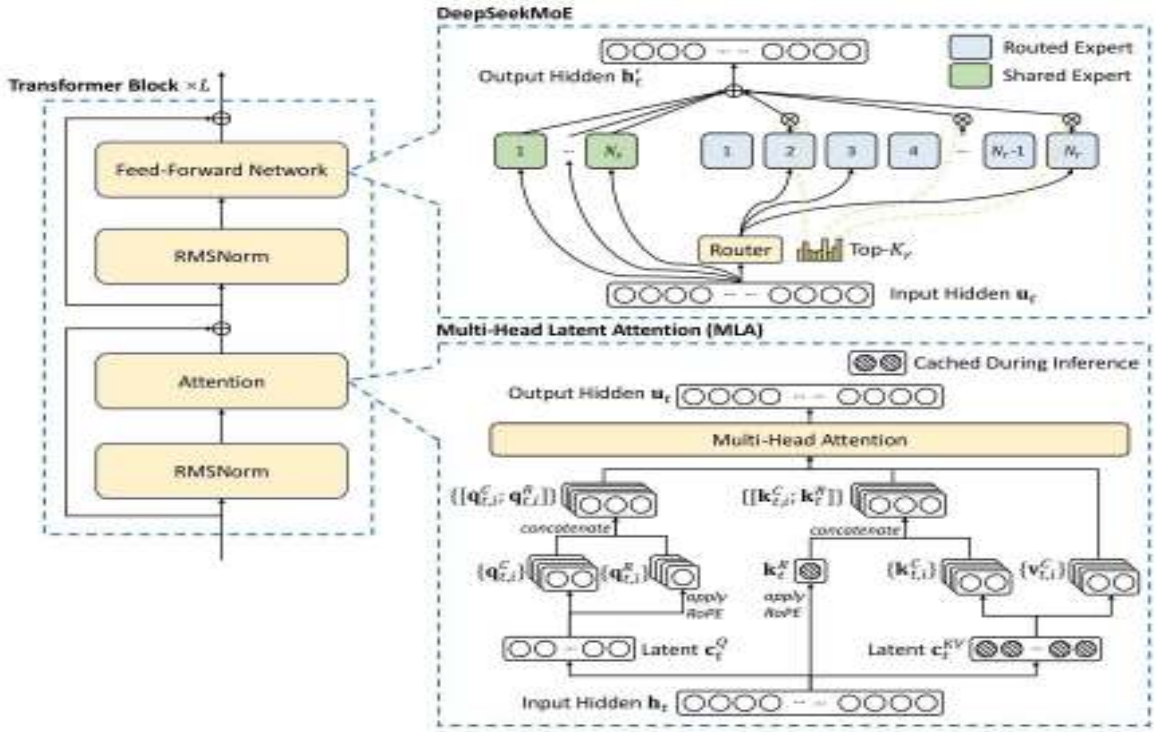
- 杭州深度求索公司开发，使命为“以开源与低成本推动AI普惠”，创始人梁文锋



DeepSeek 模型架构

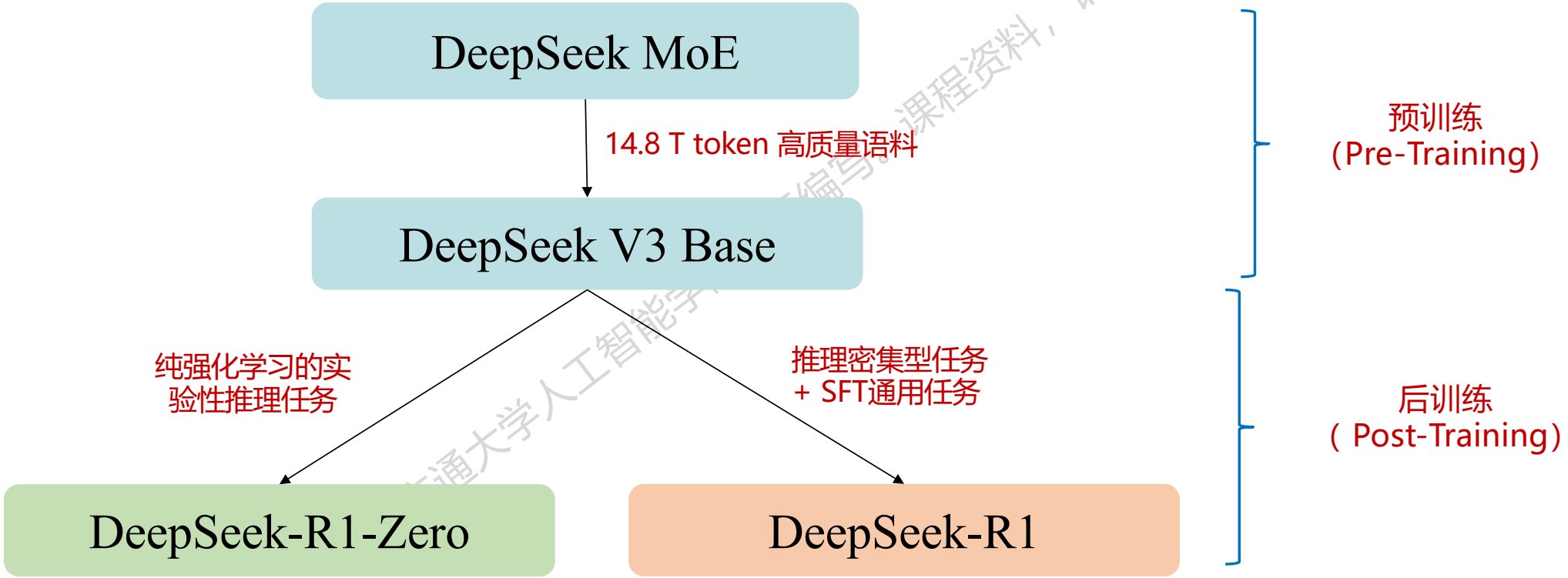
改进前馈网络与注意力机制的Transformer

- 混合专家系统DeepSeekMoE
  - MoE：每生成一个词元经路由后激活不同参数
  - DeepseekMoE：设置更多专家（细粒度），引入共享专家
- 多头隐式注意力MLA
  - KQV降维：将KQV矩阵降维到低维隐空间
  - 减少推理时KV-Cache，提高推理效率



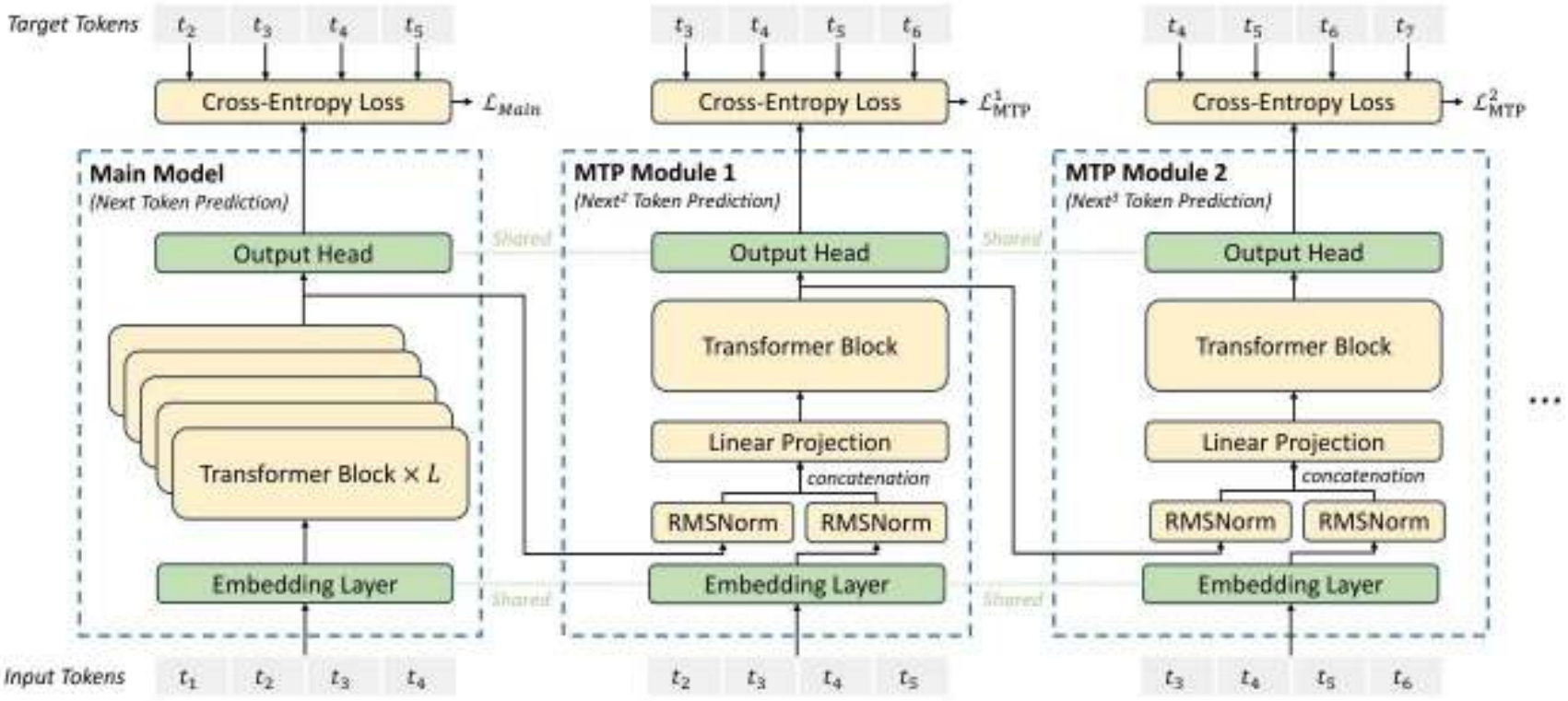
DeepSeek 模型训练

大规模预训练 + 面向推理的后训练



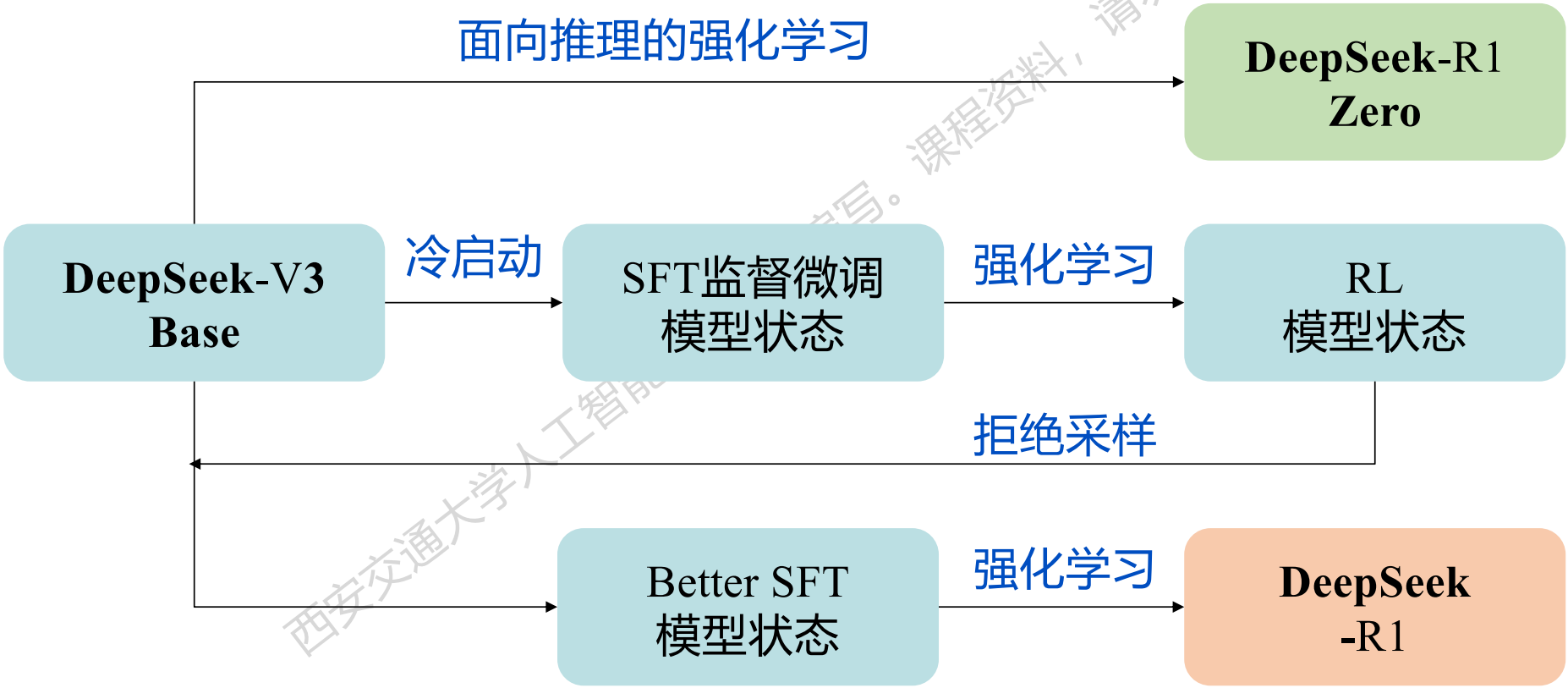
DeepSeek 预训练MTP

- 在大规模文本上进行预训练，传统方式是预测下一个token，DeepSeek设置多token预测目标函数 (MTP)，一次预测多个token，提高训练与推理效率



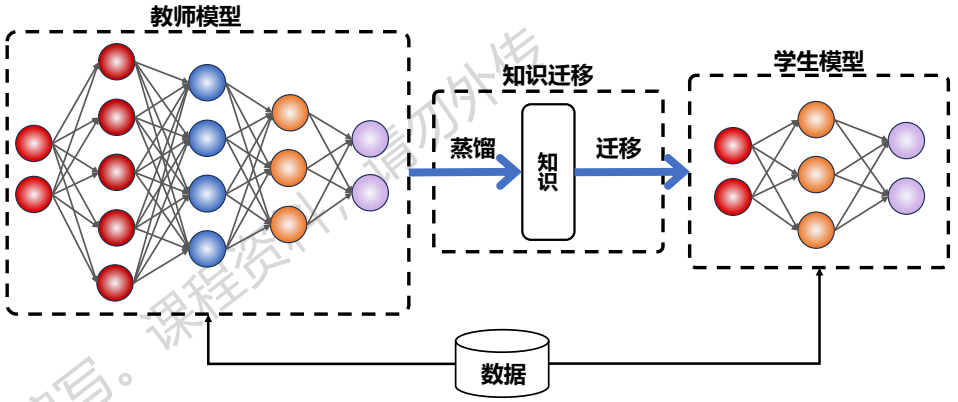
DeepSeek 面向推理的后训练

面向推理的后训练



知识蒸馏 Knowledge Distillation

- 知识蒸馏是一种将大型复杂模型（教师模型）的知识迁移到小型高效模型（学生模型）的技术，使学生模型性能更好、更好部署应用



- 使用 DeepSeek-R1 生成80万条数据，对Qwen和Llama进行微调蒸馏，在特定任务表现比更大的推理模型更优秀

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCode Bench	CodeForces
	pass@1	cons@64	pass@1	pass@1	pass@1	rating
GPT-4o-0513	9.3	13.4	74.6	49.9	32.9	759
Claude-3.5-Sonnet-1022	16.0	26.7	78.3	65.0	38.9	717
OpenAI-o1-mini	63.6	80.0	90.0	60.0	53.8	**1820**
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9	1316
DeepSeek-R1-Distill-Qwen-1.5B	28.9	52.7	83.9	33.8	16.9	954
DeepSeek-R1-Distill-Qwen-7B	55.5	83.3	92.8	49.1	37.6	1189
DeepSeek-R1-Distill-Qwen-14B	69.7	80.0	93.9	59.1	53.1	1481
DeepSeek-R1-Distill-Qwen-32B	**72.6**	83.3	94.3	62.1	57.2	1691
DeepSeek-R1-Distill-Llama-8B	50.4	80.0	89.1	49.0	39.6	1205
DeepSeek-R1-Distill-Llama-70B	70.0	**86.7**	**94.5**	**65.2**	**57.5**	1633

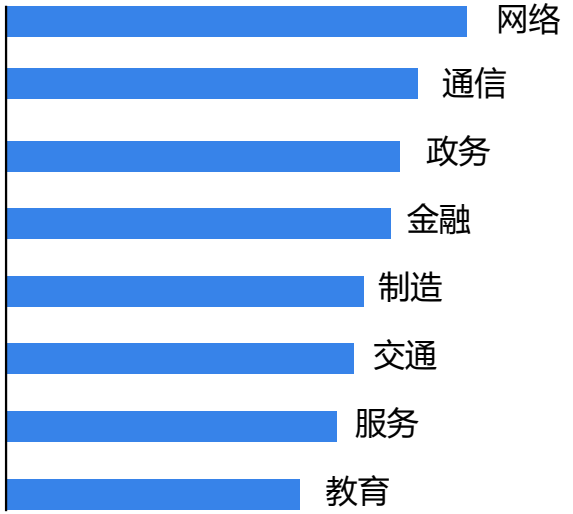
DeepSeek：先进 开源 高效 低成本

	DeepSeek V3	DeepSeek R1	OpenAI GPT-4o	OpenAI o1
发布时间	2024/12	2025/01	2024/05	2024/09
是否开源	开源	开源	闭源	闭源
上下文长度	128K	128K	128K	200K
支持模态	仅文本	仅文本	文本、图像	文本、图像
训练成本	557.6万美元	/	约1亿美元*	/
输入token价格	0.27美元/百万	0.55美元/百万	2.5美元/百万	15美元/百万

*估计值，详细数值没有公开


00101
11001
LLM
LARGE LANGUAGE MODEL

人工智能正在给人类社会各个领域带来深刻影响



我国人工智能行业渗透度排名

西安交通大学
XI'AN JIAOTONG UNIVERSITY

IAIR Est. 1986
Institute of Artificial Intelligence and Robotics

人工智能学院
College of Artificial Intelligence, XJTU

谢谢