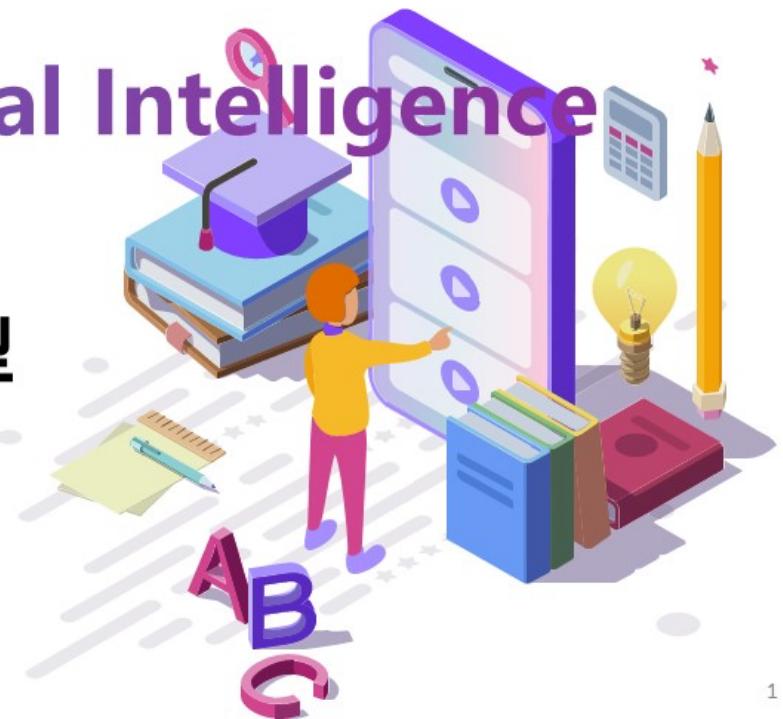


# 人工智能导论

Introduction to Artificial Intelligence

## 第六章 预训练模型与大语言模型



单选题 1分

自编码器中，编码器的作用是什么？

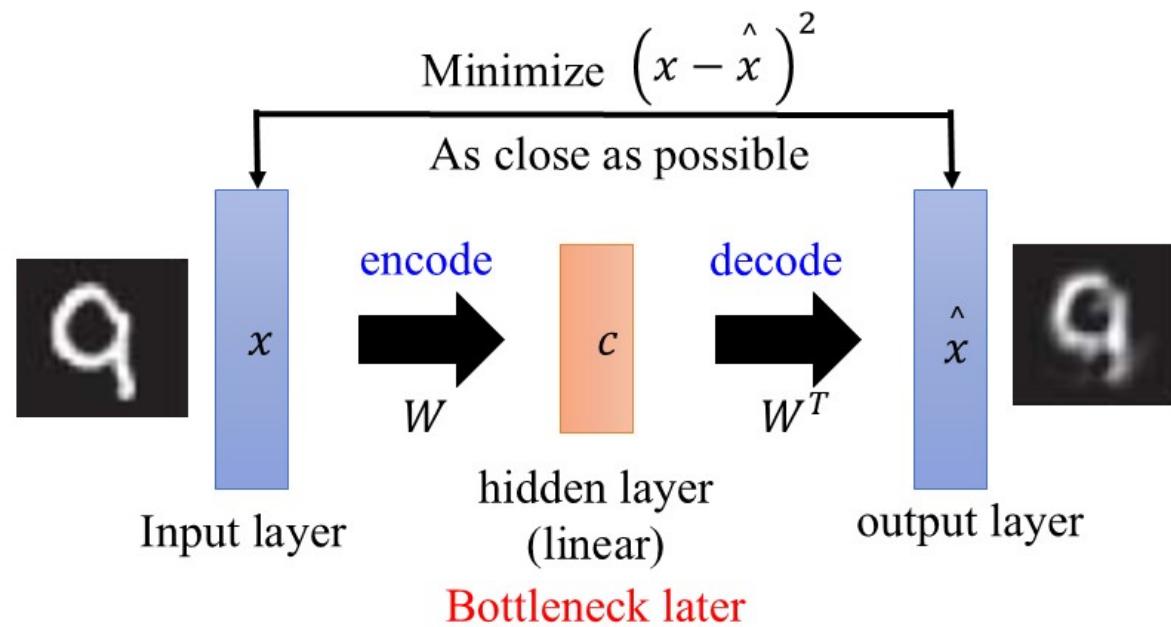
- A 将输入映射到隐空间表示
- B 从隐空间生成新样本
- C 计算损失函数
- D 为输入数据添加噪声

单选题 1分

下列哪一项最能描述自编码器（Autoencoder, AE）的核心目标？

- A 学习数据的生成分布
- B 重建输入数据，使输出尽可能接近原输入
- C 通过对抗损失生成新样本
- D 对输入数据进行分类

# AE



单选题 1分

相比普通 AE，VAE 的关键改进在于：

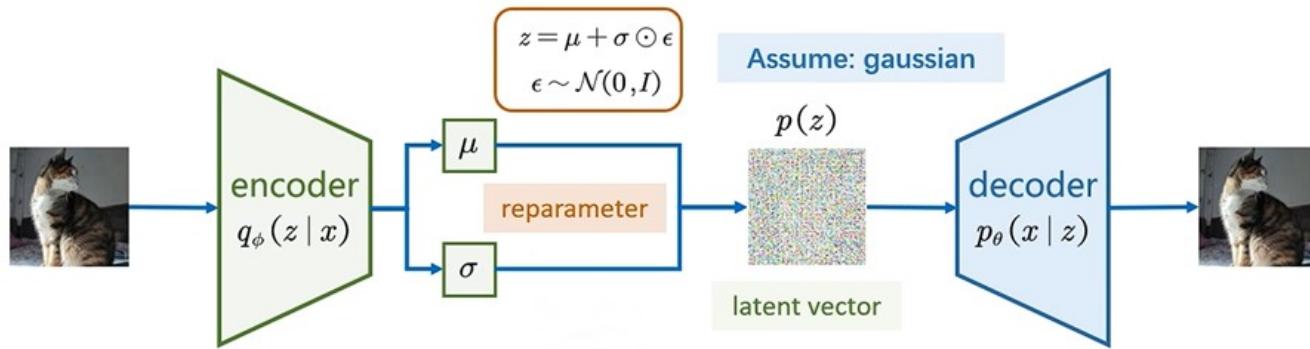
- A 使用卷积层替代全连接层
- B 在隐空间中引入概率分布假设
- C 采用对抗损失训练
- D 不再需要解码器

单选题 1分

VAE 中的 “重参数化技巧 (Reparameterization Trick) ” 的主要目的是：

- A 提高采样速度
- B 使随机采样过程可反向传播
- C 降低模型复杂度
- D 提升模型生成质量

# VAE



- **Encoder (推断网络)**：用于原始输入数据的变分推断，生成隐变量的变分概率分布情况；
- **Decoder (生成网络)**：根据生成的隐变量变分概率分布还原为原始数据近似概率分布。

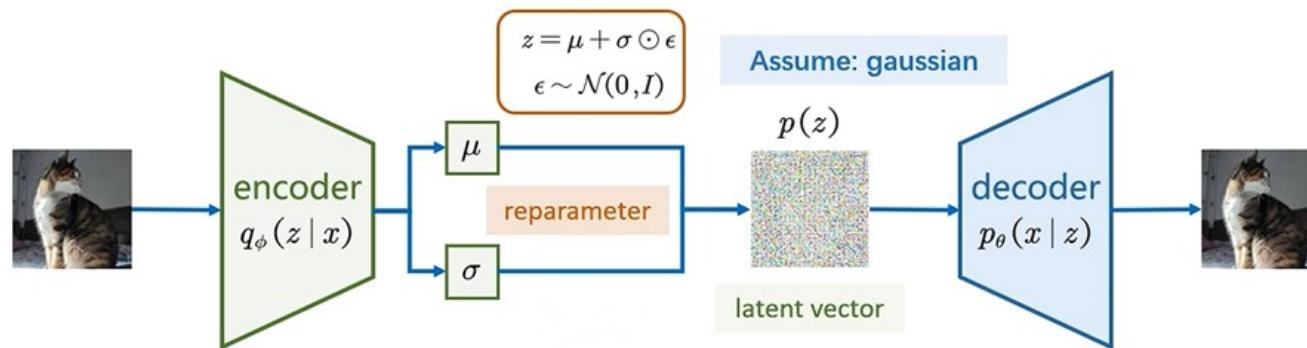
# VAE

## • 重参数化 (reparameterization)

我们需要从  $q_\theta(z|x)$  中采样  $z$ , 但采样是随机操作, 不可导, 无法通过反向传播更新参数。

$$\begin{array}{c} z \sim N(\mu, \sigma^2) \\ \xrightarrow{\hspace{1cm}} \\ \epsilon \sim N(0, I) \\ z = \mu + \sigma \odot \epsilon \end{array}$$

把采样改写成一个确定性可导函数 + 一个独立噪声项

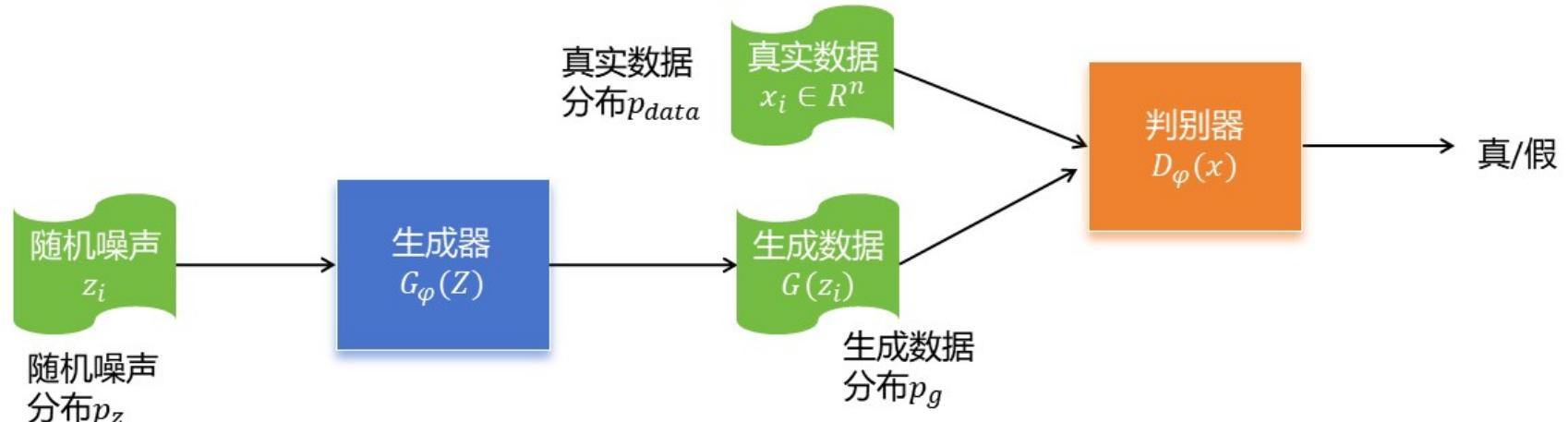


单选题 1分

GAN 的训练过程可以视为：

- A 一个优化单一目标的监督学习
- B 编码器与解码器的协同训练
- C 生成器与判别器之间的博弈
- D 多任务联合学习

# GAN



## 生成器 (Generator, G)

- 输入一个随机噪声  $z \sim p(z)$
- 输出一个伪造样本  $G(z)$ ，试图“以假乱真”
- 它的目标是：**骗过判别器**

## 判别器 (Discriminator, D)

- 输入一个样本  $(x, G(z))$
- 输出一个概率  $D(\cdot) \in [0, 1]$ ，表示“它是真实的可能性”
- 它的目标是：**识破生成器的伪造**

**对抗博弈训练：**在训练过程中交替优化判别器和生成器

单选题 1分

**扩散模型 (Diffusion Model) 的核心思想是：**

- A 从噪声逐步生成数据
- B 直接从隐空间采样
- C 使用对抗训练生成样本
- D 强化学习优化生成质量

单选题 1分

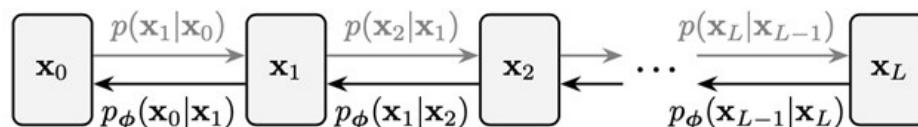
在 Diffusion 模型中，前向过程（Forward Process）与反向过程（Reverse Process）分别表示：

- A 生成新样本 / 分类样本
- B 加噪 / 去噪
- C 训练 / 推理
- D 采样 / 优化

# Diffusion

- Two Key Phases:

- **Forward Process:** Gradually add noise to data until it becomes random.
- **Reverse Process:** Learn to remove noise step by step, recovering the original data.



**Figure 2.3: Illustration of DDPM.** It consists of a fixed forward process (in gray) that gradually adds Gaussian noise to the data, and a learned reverse process that denoises step-by-step to generate new samples.

# 内容导览

- 预训练及其发展历程
- 预训练模型
- 微调技术
- 强化学习



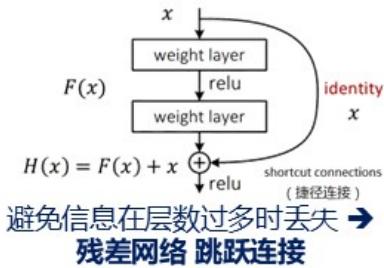
# 内容导览

## ➤ 预训练及其发展历程

- 预训练模型
- 微调技术
- 强化学习

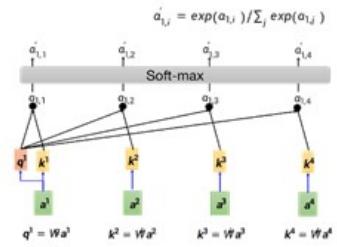


# 课程回顾



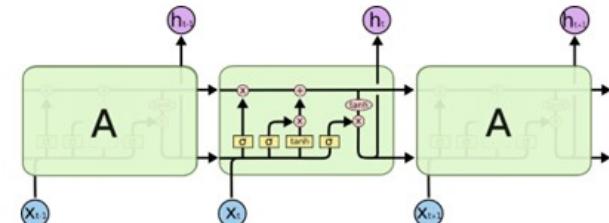
避免信息在层数过多时丢失 →  
残差网络 跳跃连接

网络架构  
关注模型的内部结构  
和信息流动方式



集中关注信息的关键部分来提取出更加重要的内容

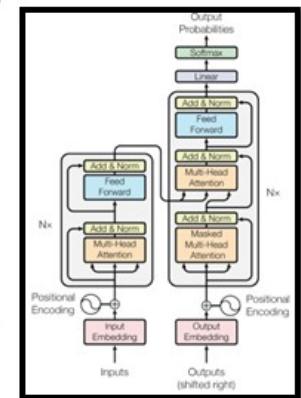
## 模型长什么样？



LSTM解决RNN长期依赖问题

“人的大脑结构”，决定你能处理什么样的信息。

解决 RNN 与 CNN 在建模长序列依赖关系时的局限性



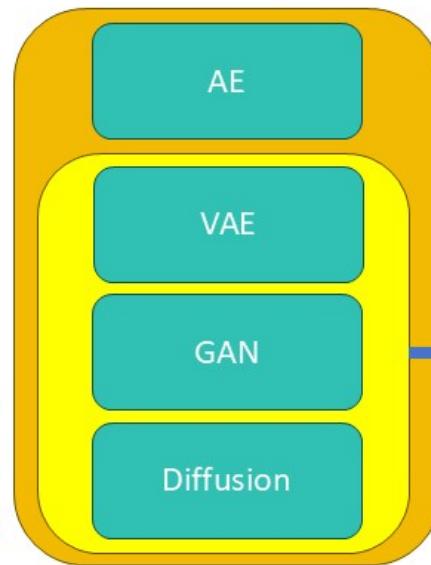
# 课程回顾

## 模型怎么学?

学习范式  
关注模型的训练目  
标和优化框架

有监督学习  
从带标签数据  
学输入-输出映  
射

无监督学习  
无监督学习提供了从  
海量无标签数据中自  
行学习规律的方法论



“学习方式”  
决定怎么学  
习知识。  
生成式  
学数据分布  
 $p(x)$ , 可生  
成

# 课程回顾

同样的大脑，选择不同的学习方式，产生不同的智能。

**Transformer** 是一个通用的  
“智能大脑” 架构。

它能处理序列、捕捉全局关系、对任意模态建模。但它的“**个性**”和“**能力**”，取决于——它是怎么学的（**学习范式**）。

**Transformer + 有监督学习**：在带标签的数据上训练，例如机器翻译，让模型成为擅长特定任务的“专家”。

- 输入：“这部电影真让人失望。”
- 输出/标签：“负面”。
- 模型通过大量标注样本学习“词语 → 情感”映射关系。

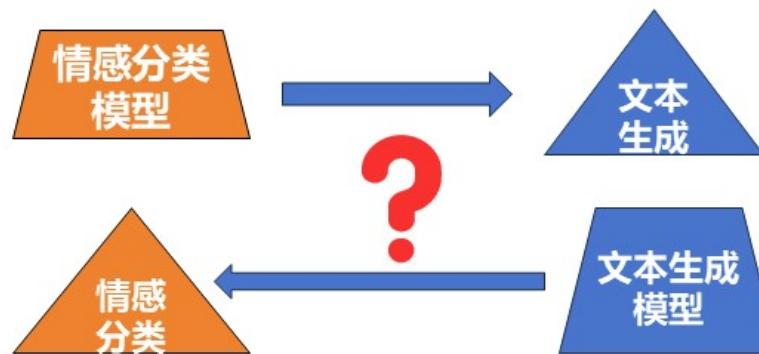
**Transformer + 生成式**：在海量无标签数据上，通过**学习数据的分布**来生成新的样本，让模型能够“理解世界的规律，并据此创造”。

- 输入：无标签文本。
- 输出：新的文本。
- 模型从大量无标注样本中学习数据分布，生成符合语言规律的新内容。

## 课程回顾



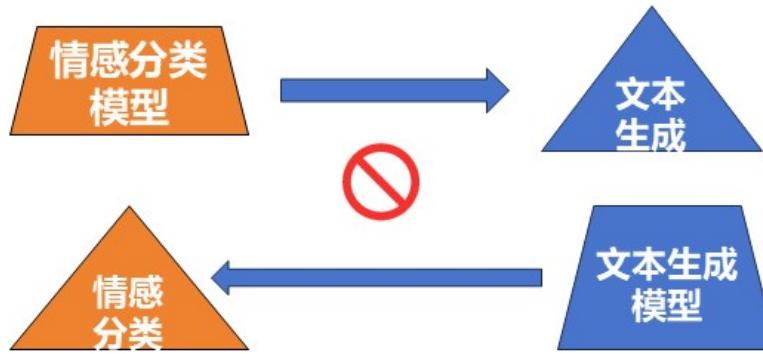
基于 Transformer 的情感分类模型，能否用来生成新的内容？  
抑或是，基于 Transformer 的生成模型，能否用  
来做分类？  
如果要实现，应该怎么做？



## 课程回顾



- 任务不同，**目标函数不同，模型参数与输出空间也完全不同。**
  - 分类模型的目标是最小化分类误差，它学习的是“判断”
  - 生成模型的目标是最大化生成概率，它学习的是“构造”。
- **彼此之间无法直接通用。**



*Training from scratch !*

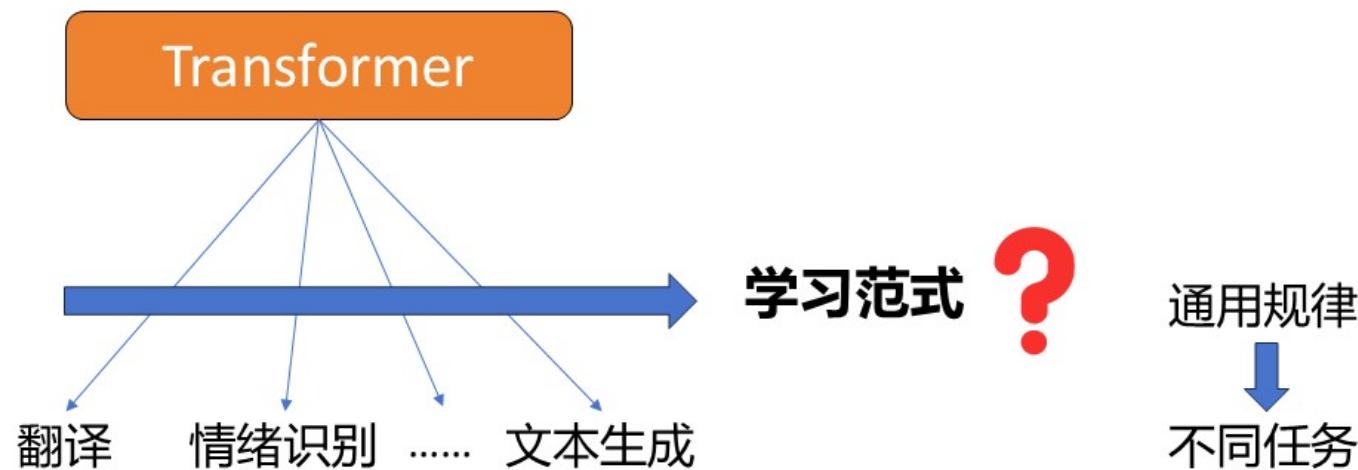


**有没有一种学习方式，能让模型先学通用规律，再适配不同任务？**

## 尝试思考



有没有一种学习方式，能让模型先学通用规律，再适配不同任务？

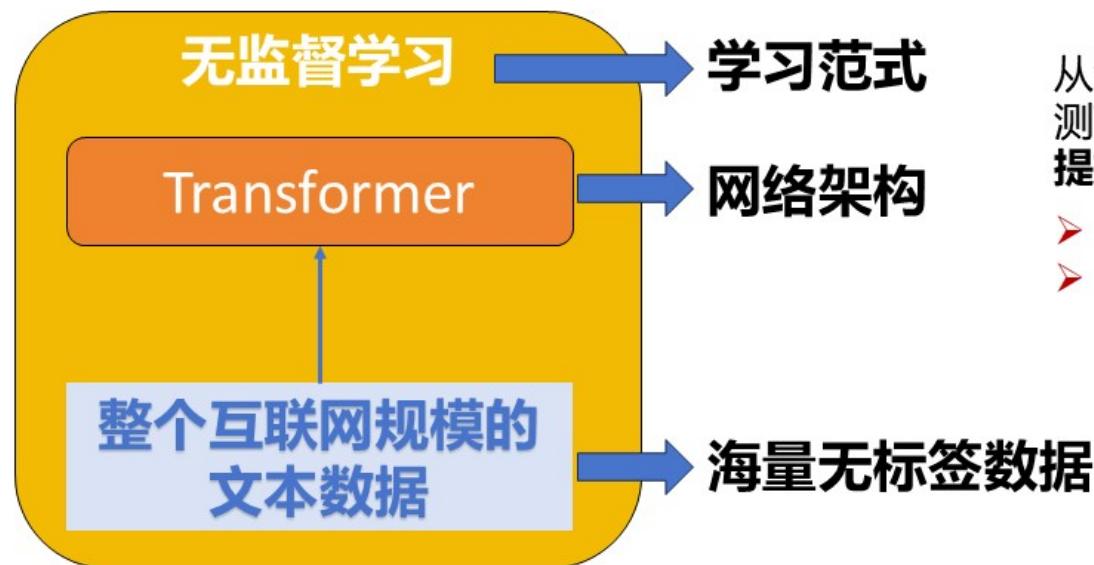


## 尝试思考



有没有一种学习方式，能让模型先学通用规律，再适配不同任务？

自回归：预测下一个词



从海量无标注数据中，模型通过“预测下一个词”的方式，**发现语言规律，提炼语义与逻辑结构。**

- **发现规律**
- **总结共性**

## 尝试思考



有没有一种学习方式，能让模型先学通用规律，再适配不同任务？



在已有通用知识的基础上，通过少量标注数据进行任务适配。

# 什么是预训练？

预训练（pre-training）是指在人工智能和机器学习中，先使用大规模、通用数据集训练模型，使其学习**通用的知识和特征**，然后再将其**迁移适配**到特定任务。

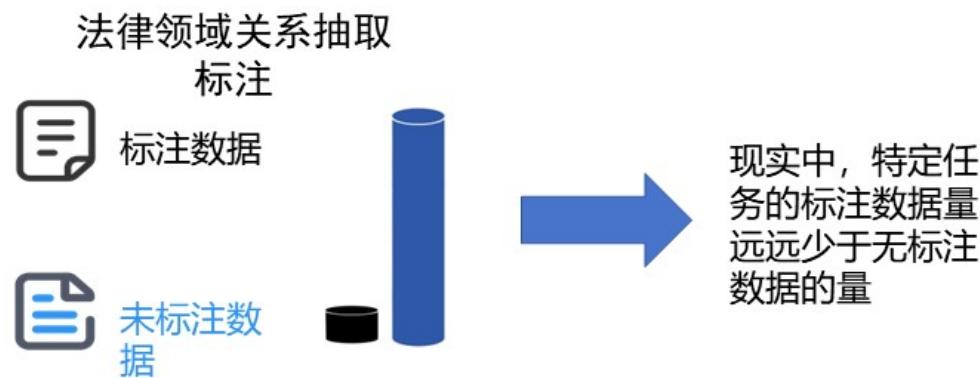
1. **发现规律，总结共性—预训练阶段**（Pre-training Phase）：模型在大规模的通用数据集上进行训练，以学习通用的知识和特征。这个阶段不针对任何特定的下游任务。
2. **迁移—微调阶段**（Fine-tuning Phase）：预训练的模型在特定任务的数据集上进行进一步训练，以适应该任务的特定需求。这个阶段的训练数据通常比预训练阶段的数据少得多。

# “预训练”解决的问题

## ➤ 标记数据资源稀缺

**标注资源稀缺而无标注资源丰富:** 某种特殊的任务只存在非常少量的相关训练数据，以至于模型**不能从中学习总结到有用的规律。**

- 许多任务依赖专家标注（医学、法律、语音）。
- 一些低资源语言、专业领域样本极少。
- 模型容易过拟合小样本，泛化性能差。



**模型缺乏“见多识广”的机会，只能在狭窄样本中死记硬背**

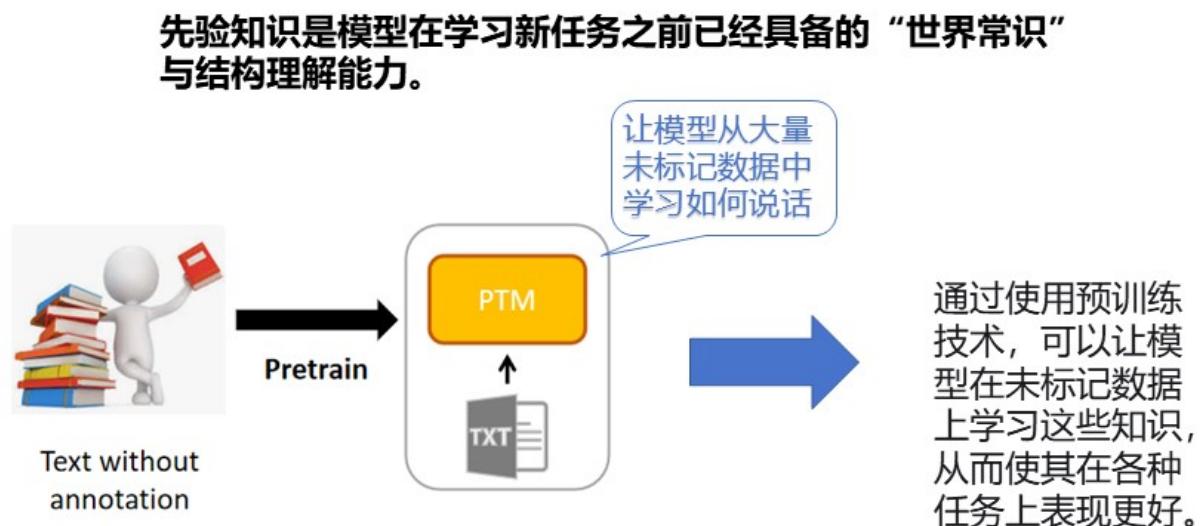
25

# “预训练”解决的问题

## ➤ 先验知识问题

**模型从零开始学习世界：**缺乏语法、语义、常识等先验知识，学习缓慢且易陷入局部最优。

- 传统神经网络参数随机初始化；
- 不懂语言结构、视觉规律、逻辑常识；
- 每个任务都要重新学习世界规律。



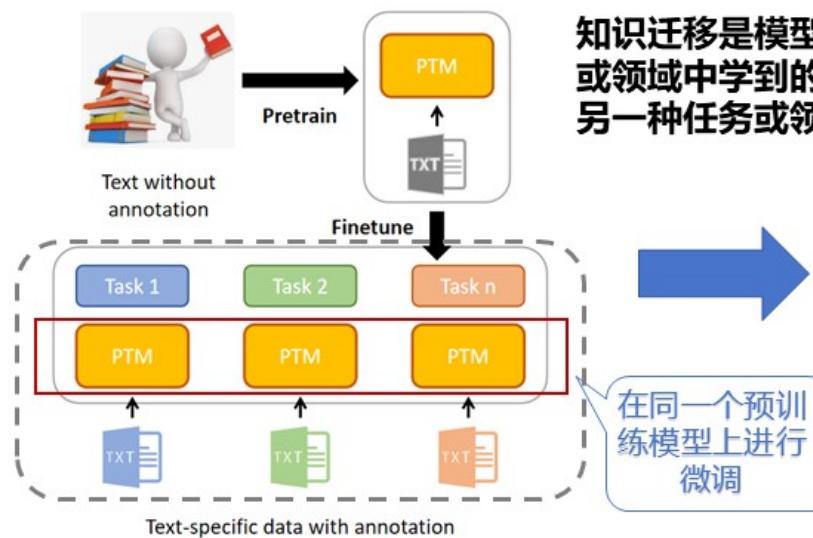
模型像“失忆的学生”，每次都得从头理解世界。

# “预训练”解决的问题

## ➤ 知识迁移问题

**任务之间缺乏迁移：**不同任务独立训练、重复造轮子，无法共享已学知识。

- 情感分析、命名实体识别、阅读理解各自独立；
- 每个任务都要重新训练模型；
- 算力浪费，且模型“学不会举一反三”。

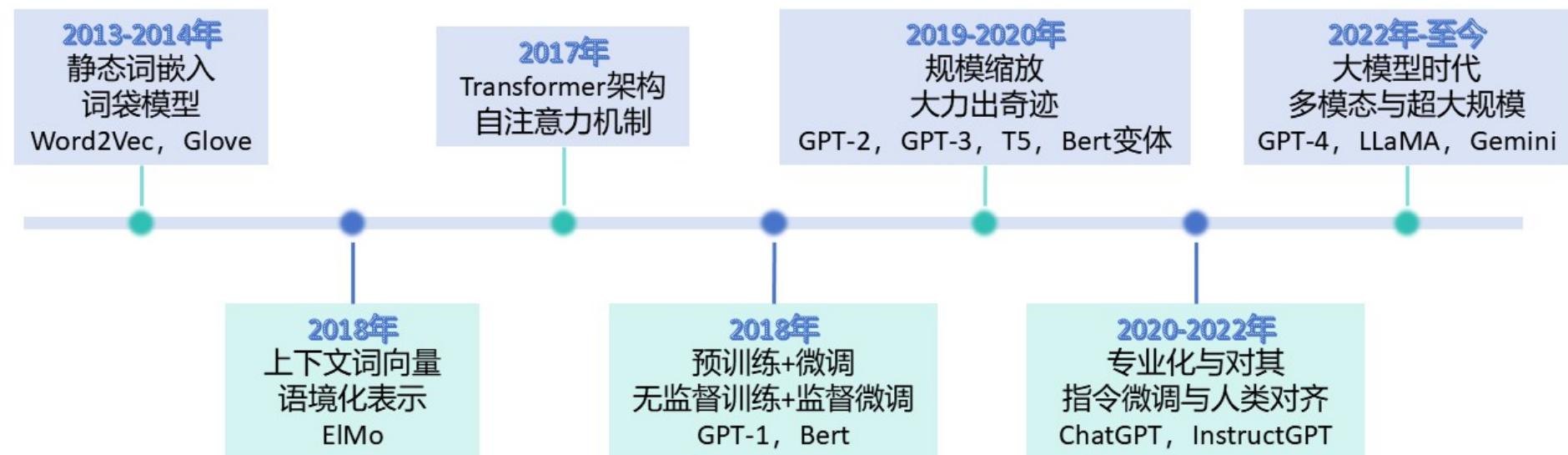


**模型无法在新任务中复用已有经验。**

27

# 预训练模型的发展历程

预训练模型的发展，是从“学词义”到“学语境”，再到“学句义、学世界”的演进过程



# 预训练语言模型的基础

## ➤ 早期探索：静态词嵌入——学语义

语言理解的第一步，是让模型学会用向量表示词的意义。

词嵌入：将每个单词或词组被映射为实数域上的向量



One-Hot将词典中所有的词排成一列，根据词的位置设计向量，例如词典中有m个词，则每个单词都表示为一个m维的向量，单词对应词典中的位置的维度上为1，其他维度为0。

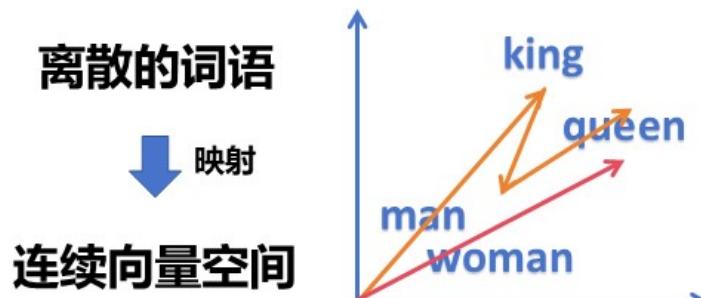
# 预训练语言模型的发展历程

## ➤ 早期探索：静态词嵌入——学语义

预训练的最初形态是**词级别的静态表征学习**，以Word2Vec为代表。

相较于 one-hot 编码，基于稠密向量的词嵌入模型能捕捉到词语之间更丰富的关系。

Word2Vec 让模型通过上下文学习词义，相似的词拥有相似的向量。



Word2Vec GloVe  
FastText

$$\text{vector}(\text{"king"}) - \text{vector}(\text{"man"}) + \text{vector}(\text{"woman"}) \approx \text{vector}(\text{"queen"})$$

向量具有“语义算术”特性

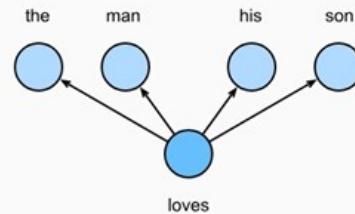
30

# 预训练模型的发展历程

## ➤ 早期探索：静态词嵌入——学语义

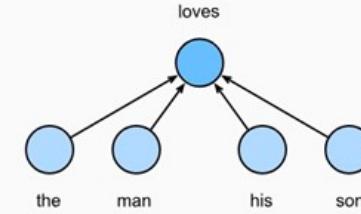
Word2vec模型包含两种结构：**Skip-gram、CBOW**

$$P(\text{"the"} \mid \text{"loves"}) \cdot P(\text{"man"} \mid \text{"loves"}) \cdot P(\text{"his"} \mid \text{"loves"}) \cdot P(\text{"son"} \mid \text{"loves"}).$$



**Skip-gram 跳元模型**

$$P(\text{"loves"} \mid \text{"the"}, \text{"man"}, \text{"his"}, \text{"son"}).$$



**CBOW 连续词袋模型**

Word2Vec 的训练任务本质是一个**上下文预测任务**。

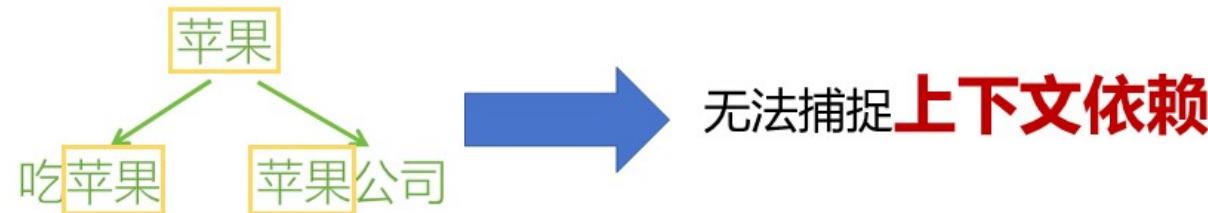
# 预训练模型的发展历程

## ➤ 早期探索：静态词嵌入——学语义

静态词向量在不同语境中含义一样，无法区分多义词。



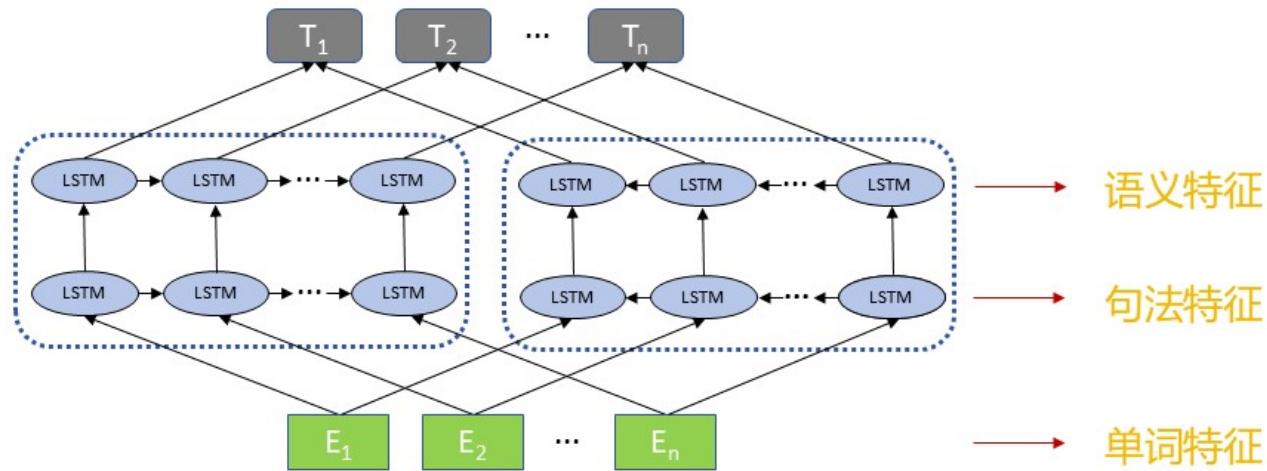
Word2Vec GloVe  
FastText



# 预训练模型的发展历程

## ➤ 上下文感知：动态词表征——学语境

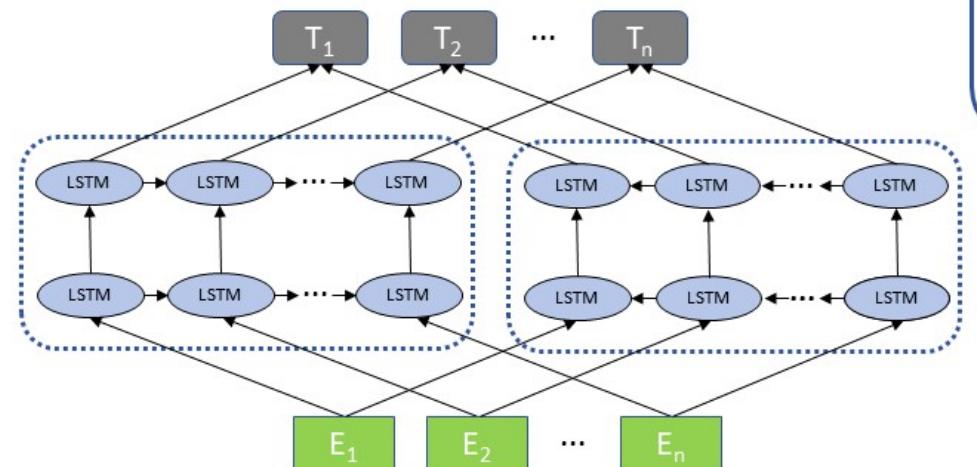
ELMo结合**RNN**循环结构的特性，把单词的上下文动态编码到单词的词向量当中。



ELMo 让词的向量不再固定，而是随上下文动态变化。

# 预训练模型的发展历程

## ➤ 上下文感知：动态词表征——学语境

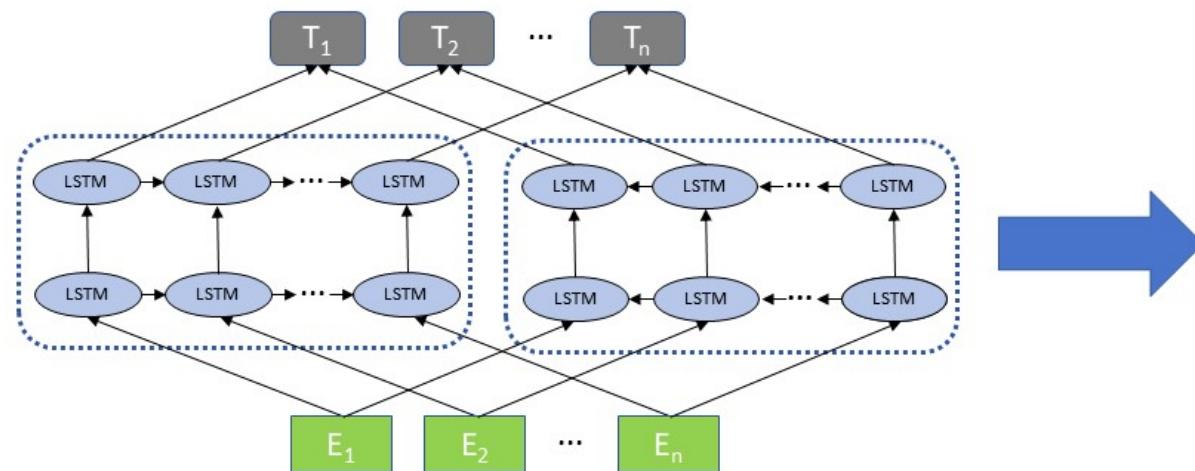


**ELMo: Embedding from language model**

- ① 双层双向LSTM，利用语言模型进行预训练
- ② 单词在不同上下文对应**不同**Embedding

# 预训练模型的发展历程

## ➤ 上下文感知：动态词表征——学语境

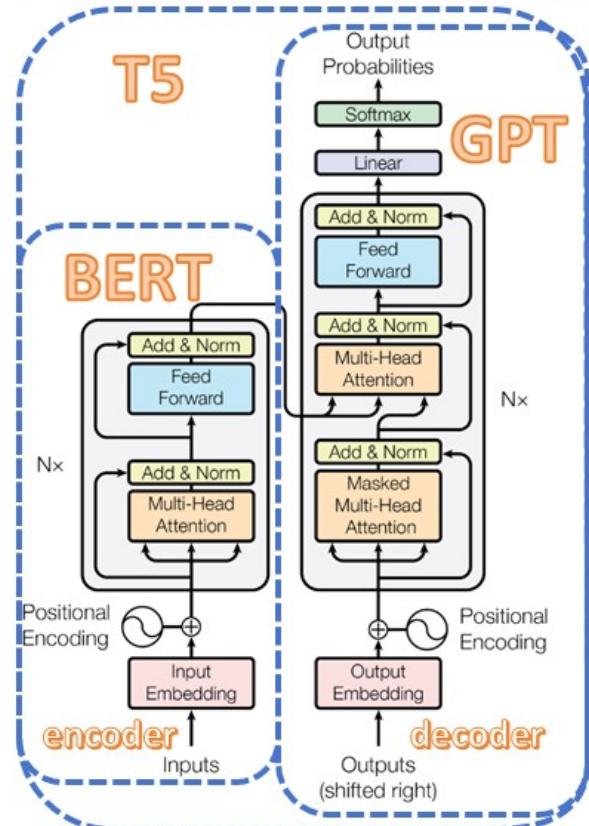


LSTM无法**并行**  
对模型的规模和性能都会有  
限制

如何让模型一次性“看到”整句话中所有词，  
并高效地捕捉它们之间的全局依赖？

# 预训练模型的发展历程

## ➤ Transformer主导：双向与通用表征——句义



Transformer 架构彻底改变了语言模型的学习方式，  
让预训练从“语境理解”迈向“句义建模”  
与“通用任务适应”。

同一架构、不同方向：GPT 生成语言，BERT 理解语言，T5 统一任务。

- **GPT:** 今天我很开心，因为…… → 自动续写。
- **BERT:** 我喜欢[MASK]。 → 填出“学习”。
- **T5:** 翻译成英文：我爱学习。 → “I love learning.”

# 预训练模型的发展历程

## ➤ 大模型时代：规模与多样化

- 模型**参数规模**步入千万亿，大模型走向对话式、生成式、多模态时代，更加注重与人类交互进行对齐，实现可靠、安全、无毒的模型。
- 各类型各系列的模型如雨后春笋，不断涌出。



GPT-4.1,  
o系列,  
...



Claude 4 Opus/Sonnet,  
Claude 4 Opus/Sonnet Extended Thinking,

...

Gemini 2.0 Flash/Gemini 2.5 Pro,  
Gemini 2.0 Flash Thinking,

...

Grok 3 (thinking)...



Mistral 3 Medium,  
Mistral Large 2,

...



Deepseek-v3,  
Deepseek-r1,  
...



GLM-4....



Doubaos-Seed-1.6,  
Doubaos-Seed-1.6-Thinking,



...

Qwen3,  
QwQ,



...

Moonshot-v1,  
kimi-latest,  
kimi-thinking-preview,



...

4.5 Turbo,  
X1,  
X1 Turbo,

...



Stable Audio

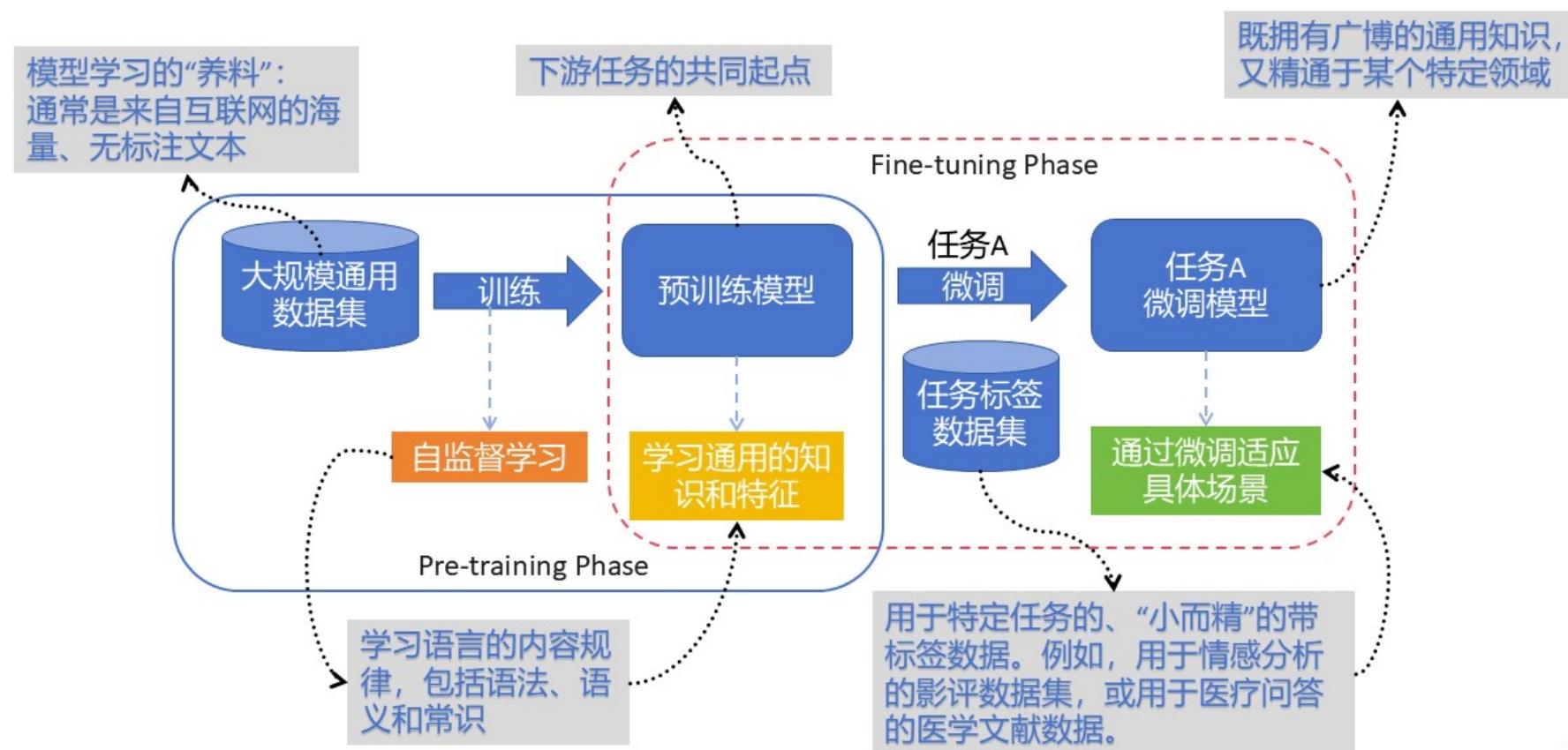
# 内容导览

- 预训练模型的发展历程
- 预训练模型
- 微调技术
- 强化学习



38

# 预训练通用范式



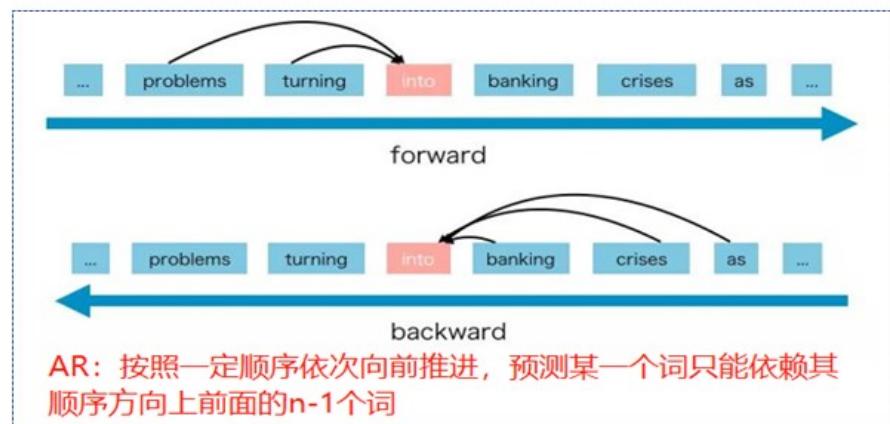
# 自监督训练

## ➤ 自监督训练

- 自监督学习是一种特殊形式的**无监督学习**，主要通过**数据本身**的内在结构来生成**标签**，而**不需要**人工标注的标签。
- 典型的自监督学习方法：
  1. **自回归式**——顺着写下去
  2. **对比式**——找出配对关系
  3. **重建式**——补全缺失内容

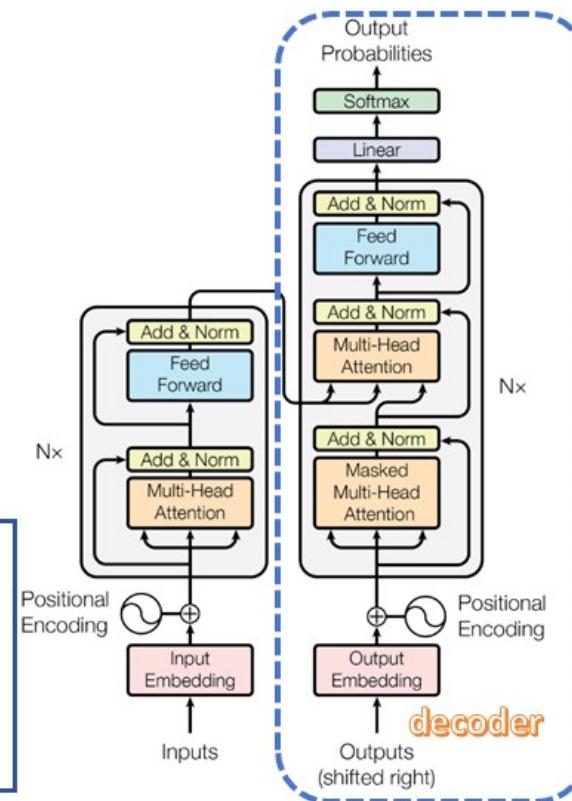
# 自监督训练

➤ **自回归：**将数据序列的生成过程视为一个链式规则，让模型学习根据前面出现的所有内容，来预测序列中下一个元素。



## 自回归式的典型方法与任务：

GPT系列是所有自回归模型的典范。它们使用Transformer Decoder架构，其核心的掩码自注意力机制确保了在预测下一个词时，只能看到它左边的上下文。还有传统的N-gram语言模型和RNN语言模型。



# 自监督训练

➤ **对比式：**不关心数据的重建，而关心数据的表示。核心是学会“比较”，拉近相似（正样本）数据的表示，推远不相似（负样本）数据的表示，从而学习一个具有高度区分度的表示空间。

## 预测式的典型方法与任务：

实例判别：将同一张图片的两个随机增强版本视为正样本对，将批次中其他所有图片视为负样本。

SimCLR, MoCo：是这一范式的经典实现，通过一个对比损失函数来明确推远负样本。

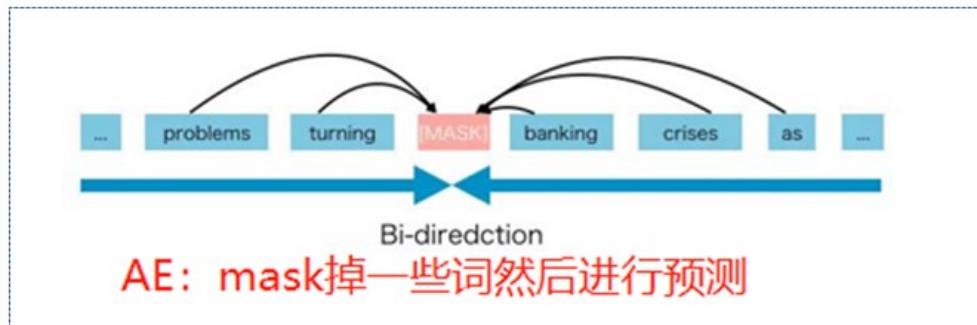


通过比较正例与负例，学习相似与不相似的关系。

42

# 自监督训练

➤ **重建式**: 让模型学习恢复原始的、完整的或受损的输入数据。模型必须在理解数据内容和结构的基础上，才能完成高质量的重建。

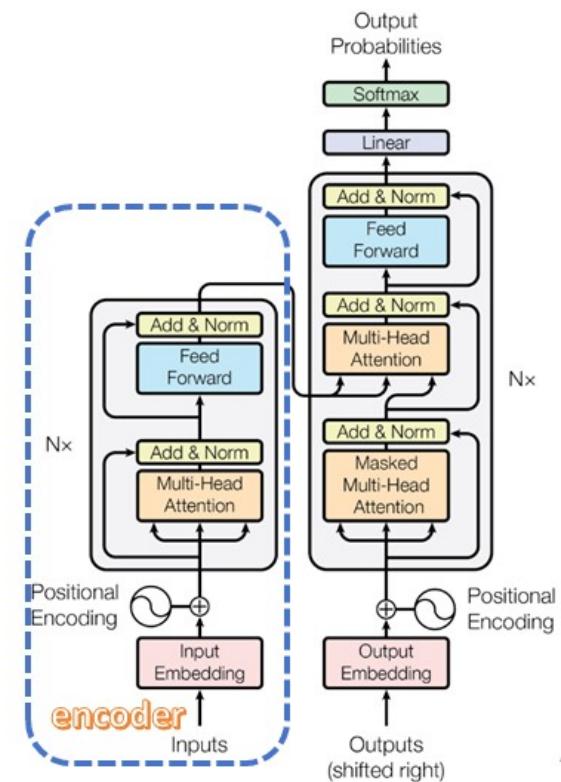


### **重建式的典型方法与任务：**

自编码器：将输入压缩再还原，目标是让输出尽可能接近输入。

**去噪自编码器**: 向输入添加噪音, 然后让模型恢复出干净的原图。

掩码建模：BERT的掩码语言模型，遮盖句子中的部分词，让模型预测被遮盖的词。



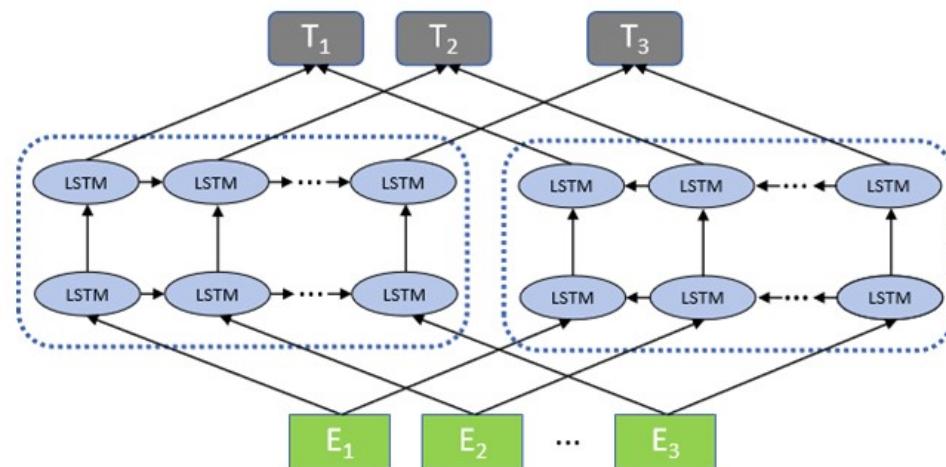
单选题 1分

ELMo属于哪一种预训练语言模型的基本范式？

- A 自回归模型
- B 重建式
- C 不属于预训练模型

# ELMo

**ELMo**虽然引入**双向语言模型**，其实是2个**单向语言模型**（前向和后向）的集成，本质上仍为**自回归语言模型**，只能获取单向的特征表示，不能同时获取上下文表示。



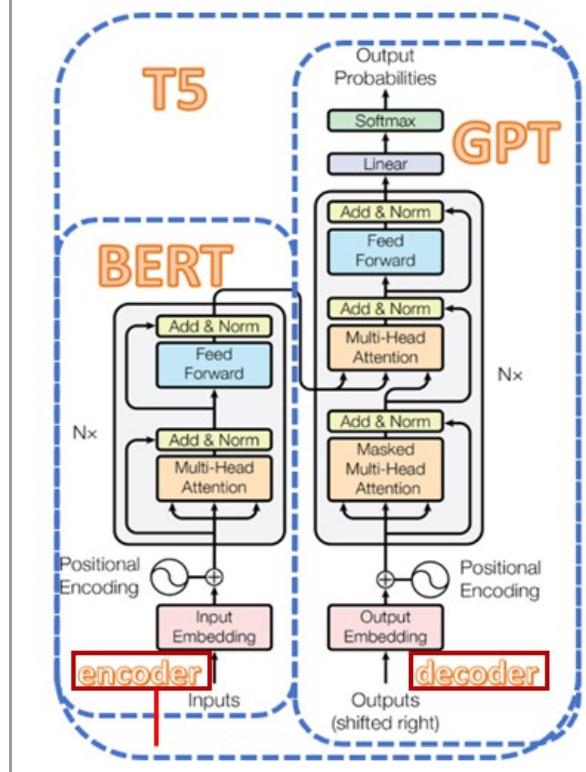
**句子:** "The cat sat on the mat."

**前向LM:** 预测下一个词 → The → cat → sat → on → the → mat

**后向LM:** 预测上一个词 → mat → the → on → sat → cat → The

# 经典的预训练语言模型

## ➤ GPT、BERT & T5：最经典的预训练模型



- **GPT**取自Transformer的**解码器**，采用生成式预训练，具有自回归特性。
- **BERT**取自Transformer的**编码器**，采用掩码语言模型和下一句预测等任务进行预训练，不具有自回归特性。
- 除此之外还存在**编码器-解码器**类型的大模型，例如**T5**。

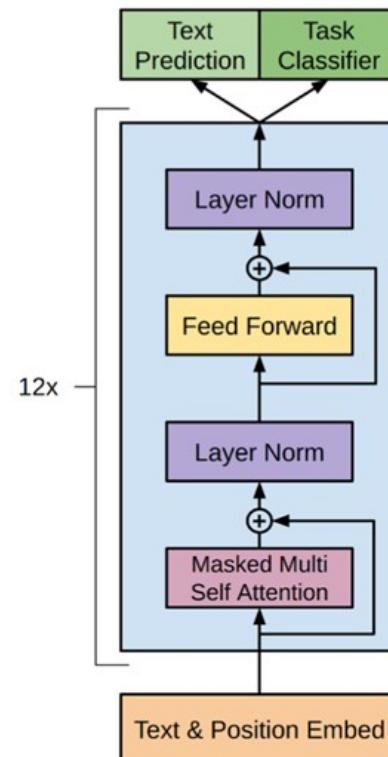
# 预训练语言模型：GPTs

## ➤ GPTs：预训练 + 微调范式的开端

在GPT-1之前，主流NLP依赖于大量任务**特定的标注数据**和**精心设计的任务特定架构**。

- 不能通用：不同损失函数在各个任务上表现差异大，训练数据集并没有包含各个NLP任务。
- 不统一：将预训练语言模型迁移到下游任务的方法不统一，不同的子任务，有时还需要调整模型结构。

GPT 之前的 NLP 模型像是“一题一解”的专家系统，  
缺乏统一框架。



## 预训练语言模型：GPTs

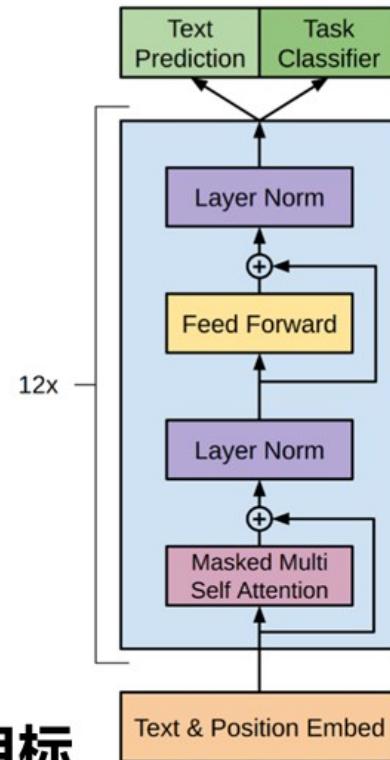
### ➤ GPTs: generative pre-trained transformer:

在GPT-1之前，主流NLP依赖于大量任务**特定的标注数据**和**精心设计的任务特定架构**。

- 不能通用：不同损失函数在各个任务上表现差异大，训练数据集并没有包含各个NLP任务。
- 不统一：将预训练语言模型迁移到下游任务的方法不统一，不同的子任务，有时还需要调整模型结构。



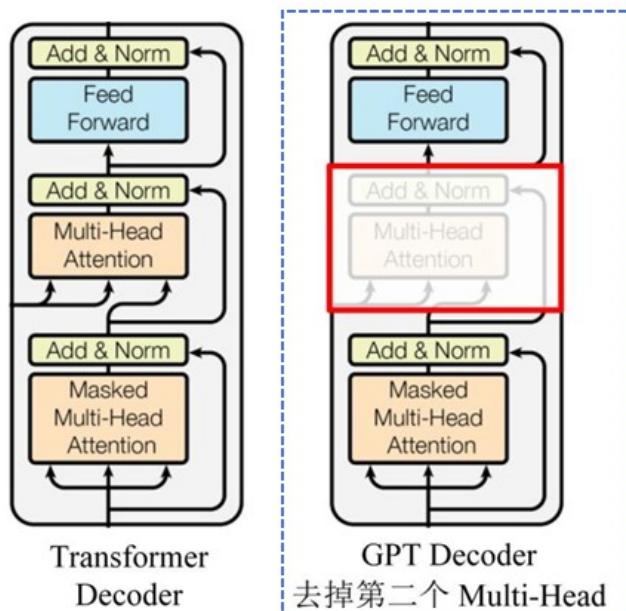
OpenAI采用一个**统一的模型架构**和一个**统一的学习目标**



# 预训练语言模型：GPTs

## ➤ GPTs: generative pre-trained transformer:

GPT采用的是以Transformer解码器为基础的生成式预训练模型。



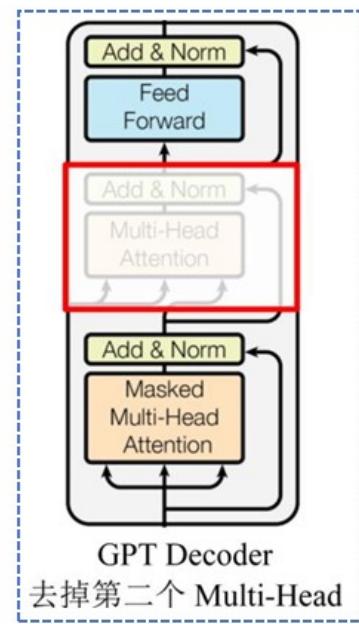
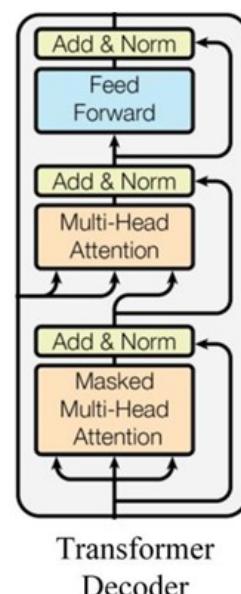
- 基于 Transformer 的 Decoder 架构 (单向)
- 只使用自注意力机制中的下三角掩码 (Mask)，保证每个词只看到它前面的词。
- 采用自回归训练

$$P(x) = \prod_t P(x_t | x_{<t})$$

将层数扩展到12层，GPT-1 还将Attention 的维数扩大到768 (原来为512) , 将 Attention 的头数增加到12层 (原来为8层) , 将 Feed Forward 层的隐层维数增加到3072 (原来为2048) , 总参数达到1.5亿。

# 预训练语言模型：GPT1

## ➤ GPT-1自监督预训练：

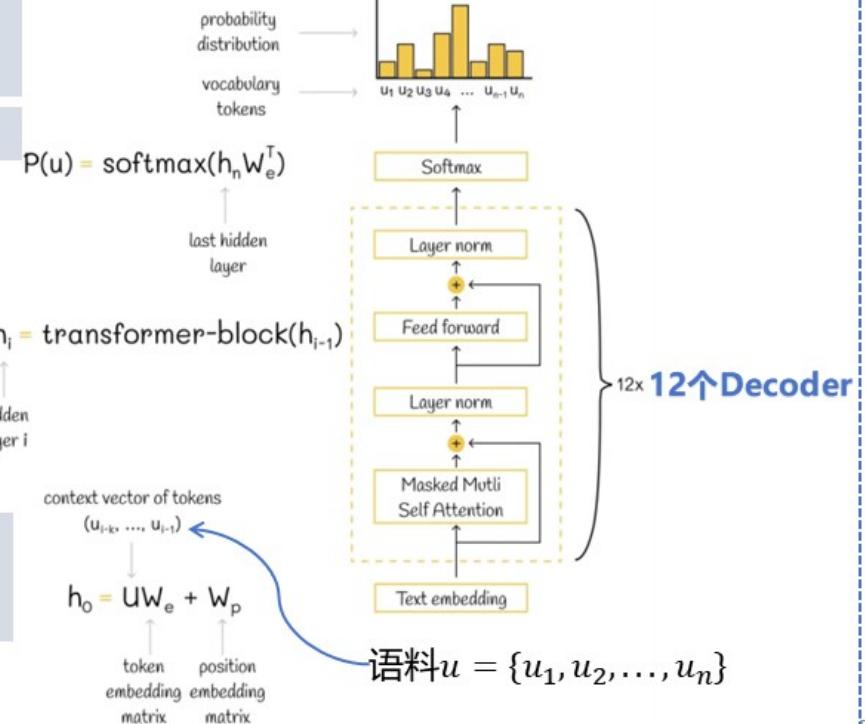


GPT1在位置编码上采用随机初始化，随训练更新。

预训练任务：  
输出下一个词的概率分布函数  
常用于文本生成

让模型看到前面k个词，然后预测下一个词

扩大网络规模 总参数达1.5亿



# 预训练语言模型：GPT1

## ➤ GPT-1分词法：Byte Pair Encoding (BPE)

### ➤ 什么是 BPE?

- 一种**基于频率合并的子词分词算法**
- 从字符级开始，反复将**出现频率最高的符号对**合并为一个新的 token

### ➤ 工作过程

1. 初始化为所有字符
2. 统计相邻符号对的出现频率
3. 合并最高频的符号对
4. 重复直到词表大小达到上限

词：internationalization

• 初始：internationalization

• 高频对（假设在语料中出现多）：

• ("i", "n") → "in"

• ("t", "i", "o", "n") → "tion"

• ("a", "l") → "al"

• 合并后：inter nation al ization

• 最终切分：[inter, nation, al, ization]

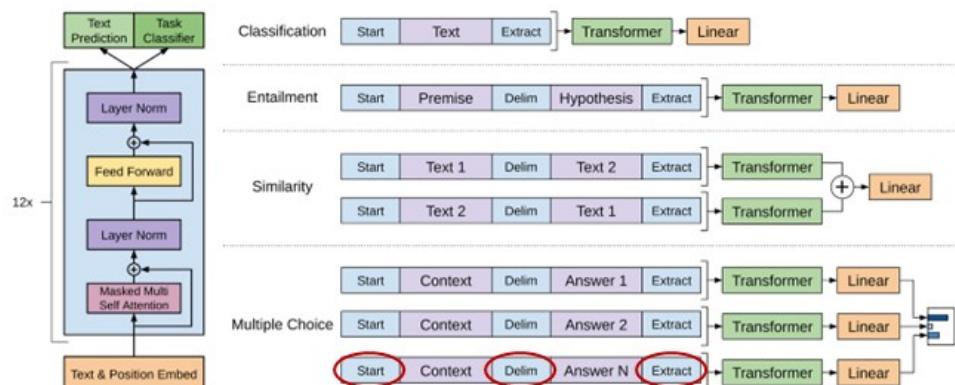
**优点：**压缩词表，能共享词根；对英文生成任务非常高效；可生成未登录词

**缺点：**对罕见词可能切得过碎；不适合形态复杂或多语言文本

# 预训练语言模型：GPT2

## ➤ GPT-2：大力出奇迹

使用更多的数据，增加模型参数量 (1.17亿 → 15亿)



GPT1引入了许多特殊token进行下游任务微调  
训练范式：自监督预训练+有监督微调

OpenAI假设

“一个在足够大和多样化的数据集上训练的、拥有足够多参数的语言模型，将能够学习到解决许多任务所需的能力，而无需任务特定的训练数据。”



GPT2的训练范式  
自监督预训练+Zero-shot learning

# 预训练语言模型：GPT2

- GPT-2：自监督预训练+zero-shot learning
- 随着模型参数量的增加
- 研究者观察到GPT-2在预训练过程中能够学习有监督任务的信息。



- 任务被构造成一段“上下文” [任务描述]，让模型去自然地补全。

input: Today is a [翻译成中文]

靠提示完成任务

output: 今天是一个

模型从给出的例子中学习到了这个任务的格式和意图

# 预训练语言模型：GPT3

## ➤ GPT-3: In-context learning

- 上下文 (in-context) 学习：在被给定的几个任务示例或一个任务说明的情况下，模型应该能通过简单预测来补全任务中的其他示例。**上下文学习要求预训练模型要对任务本身进行理解。**
- 研究者探索了**"固定提示+少样本示范"**的策略。

GPT2→GPT3

在下游任务上从零样本扩展到少样本

- Zero-shot、One-shot、Few-shot

$$L = - \sum_t \log P(x_t | x_{1:t-1})$$

GPT-3 证明了语言模型的能力与规模呈幂次增长，是“Scaling Law”的实证模型。

# 预训练语言模型：GPT3

## ➤ GPT-3: In-context learning

### Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

- 1 Translate English to French: ← task description
- 2 cheese => ← prompt

### One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

- 1 Translate English to French: ← task description
- 2 sea otter => loutre de mer ← example
- 3 cheese => ← prompt

### Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

- 1 Translate English to French: ← task description
- 2 sea otter => loutre de mer ← examples
- 3 peppermint => menthe poivrée
- 4 plush girafe => girafe peluche
- 5 cheese => ← prompt

# 预训练语言模型：GPT3

## ➤ GPT-2&3的位置编码

从 Learned Absolute 到 Rotary Position Embedding (RoPE)。

RoPE 的目标是显式建模相对位置差，而不是绝对编号。

给定查询向量  $q_m$ 、键向量  $k_n$ ，RoPE 将它们按位置旋转：

$$\text{RoPE}(q_m) = R_m q_m, \quad \text{RoPE}(k_n) = R_n k_n$$

注意力计算变为：

$$A_{mn} = (R_m q_m)^T (R_n k_n)$$

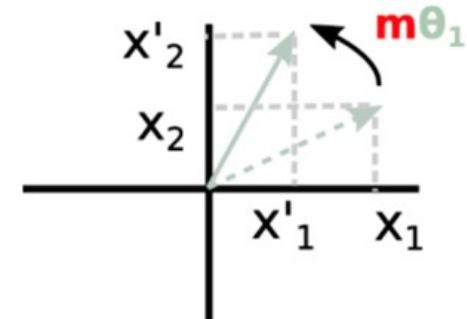
而旋转矩阵  $R_m$  定义为：

$$R_m = \begin{bmatrix} \cos(\theta_m) & -\sin(\theta_m) \\ \sin(\theta_m) & \cos(\theta_m) \end{bmatrix}$$

其中  $\theta_m = m / 10000^{2i/d_{model}}$ 。

- 更自然地捕捉“距离”信息；
- 对长序列泛化性强；
- 支持动态扩展上下文长度 ( $>8k$  tokens)。

在做 self-attention 之前，会用词嵌入向量计算 QKV 向量同时加入位置信息

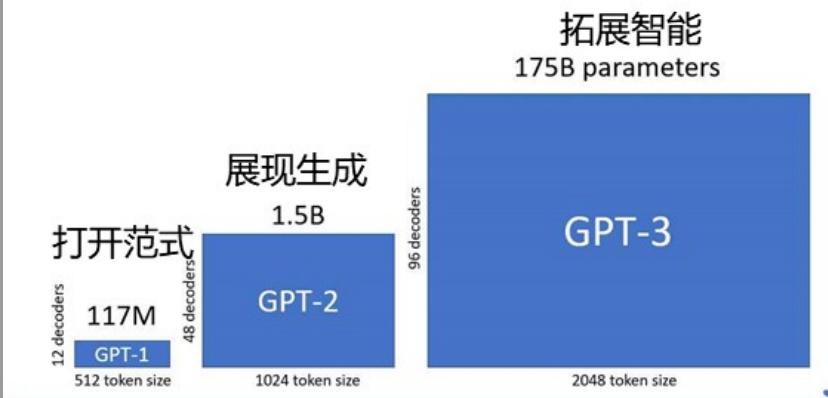


# 预训练语言模型：GPTs

## ➤ GPTs：从学习语言到展现智能

规模扩展方向：**↑层数   ↑维度   ↑数据   ↑参数量**

当模型规模、数据量、计算量持续增长时，语言模型的性能会以可预测的幂次规律稳定提升。



随着**模型参数的增多**，研究者发现了自监督预训练的GPT逐渐具有了处理需要有监督微调的下流任务的能力。

研究者尝试了以下方法来挖掘GPT的这种**涌现能力**：

- 在下游任务中嵌入**固定提示**的零样本微调；
- 嵌入**提示+演示**(称作上下文学习)的少样本微调。

这些能力在小规模模型中不存在，而在大規模模型中存在

## 预训练语言模型：BERT

- **BERT: Bidirectional Encoder Representation from Transformers**

让模型真正理解上下文，是“理解式”预训练模型的代表。

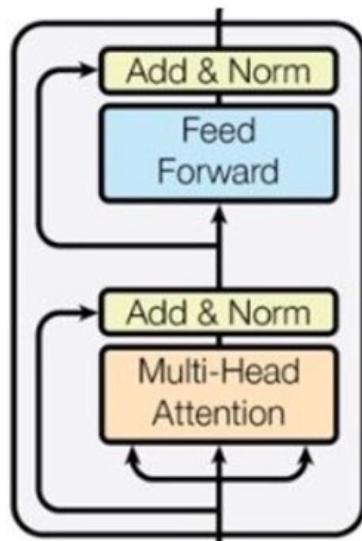
- GPT 是**单向的**，只能从左到右预测下一个词。
- ELMo 虽然**双向**，但两个方向独立训练，无法充分交互。

我们能不能让模型同时看到前后文，真正理解整句话？

# 预训练语言模型：BERT

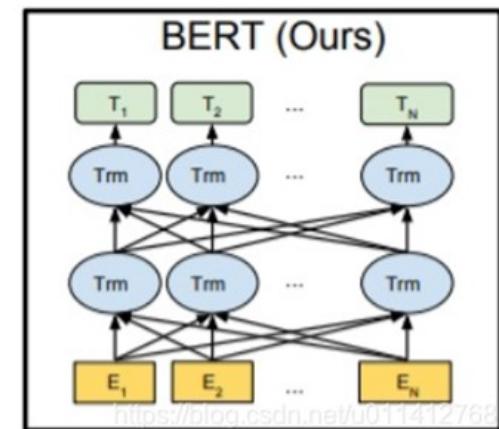
## ➤ BERT

BertLayer (Encoder Block)



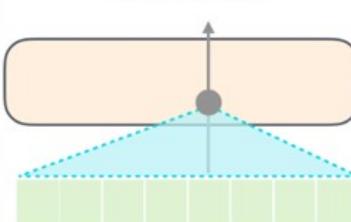
➤ 基于 **Transformer Encoder** (全双向自注意力) 结构。

➤ BERT 是“双向”——每个词都能同时看到前后文

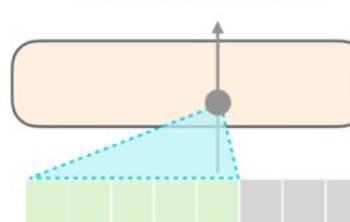


https://blog.csdn.net/01141268

Self-Attention



Masked Self-Attention



59

# 预训练语言模型：BERT

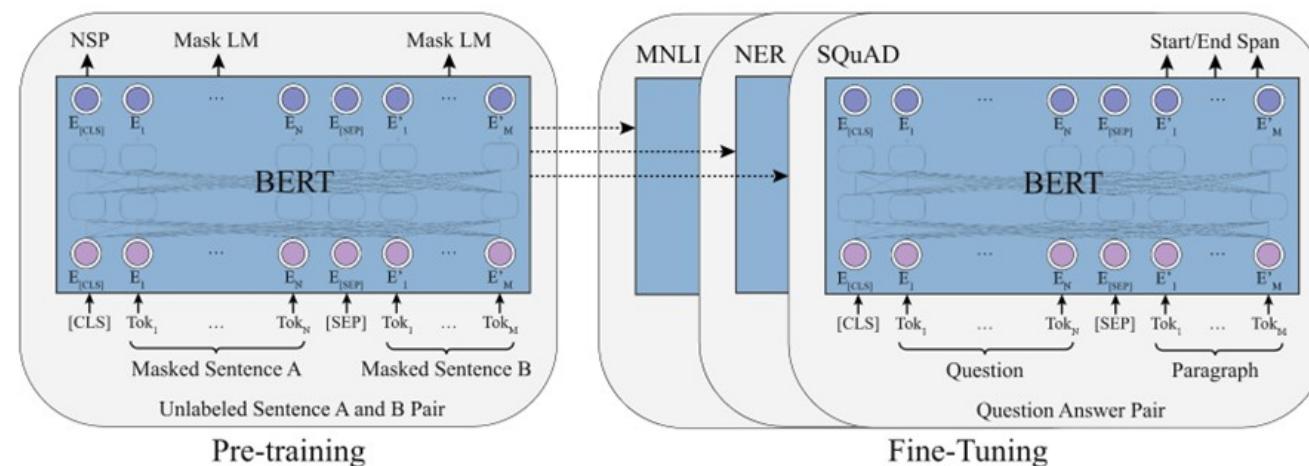
## ➤ BERT

和 GPT 一样，BERT 也采用二段式训练方法：

- 第一阶段：使用易获取的大规模无标签语料，来训练基础语言模型；
- 第二阶段：根据指定任务的少量带标签训练数据进行微调训练。

预训练任务：  
语言掩码模型（MLM）  
训练词的语义理解能力  
下句预测（NSP）  
训练句子之间的理解能力

常用于文本理解，生成  
词嵌入用于后续任务



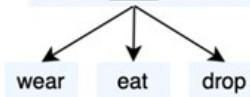
# 预训练语言模型：BERT

## ➤ MLM: 完形填空

传统的语言模型是以word2vec为代表的词向量模型，他们主要是以词袋（N-Gram）为基础。

Input: It is very cold today, we need to wear more clothes

N-Gram: It is very cold today, we need to \_\_\_\_\_ more clothes

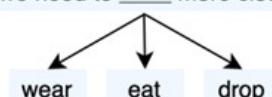


被截断的短文本

VS

Input: It is very cold today, we need to wear more clothes

MLM: It is very cold today, we need to \_\_\_\_\_ more clothes



完整的文本

替换策略 ?      训练目标 ?

# 预训练语言模型：BERT

## ➤ MLM 替换策略

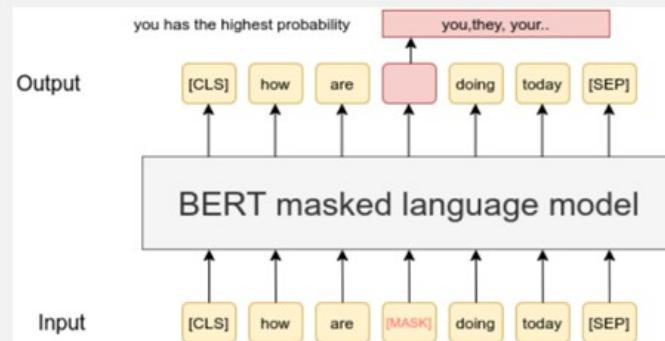
- **随机选择 15%：**比例太低，训练成本高且样本效率低；比例太高，上下文信息不足，任务太难，模型无法有效学习。
- **对 15% 的 token 进行 80-10-10% 的替换：**

- 1) 80% [MASK]
- 2) 10% 随机的其他token
- 3) 10% 原来的token

my dog is hairy——>my dog is [MASK]  
my dog is hairy——>my dog is apple  
my dog is hairy——>my dog is hairy

## ➤ MLM 训练目标

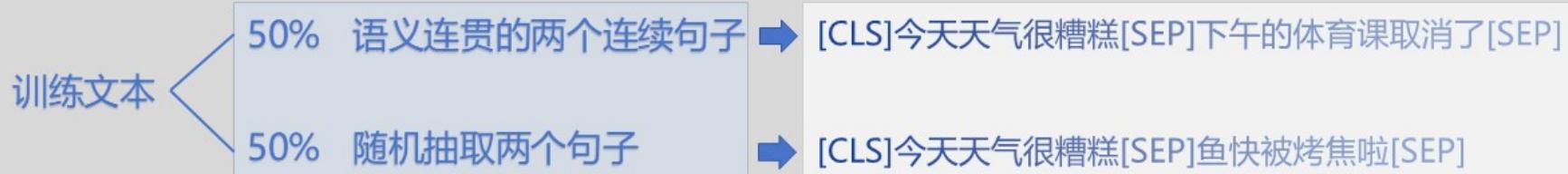
- 完形填空



# 预训练语言模型：BERT

## ➤ NSP: Next Sentence Prediction

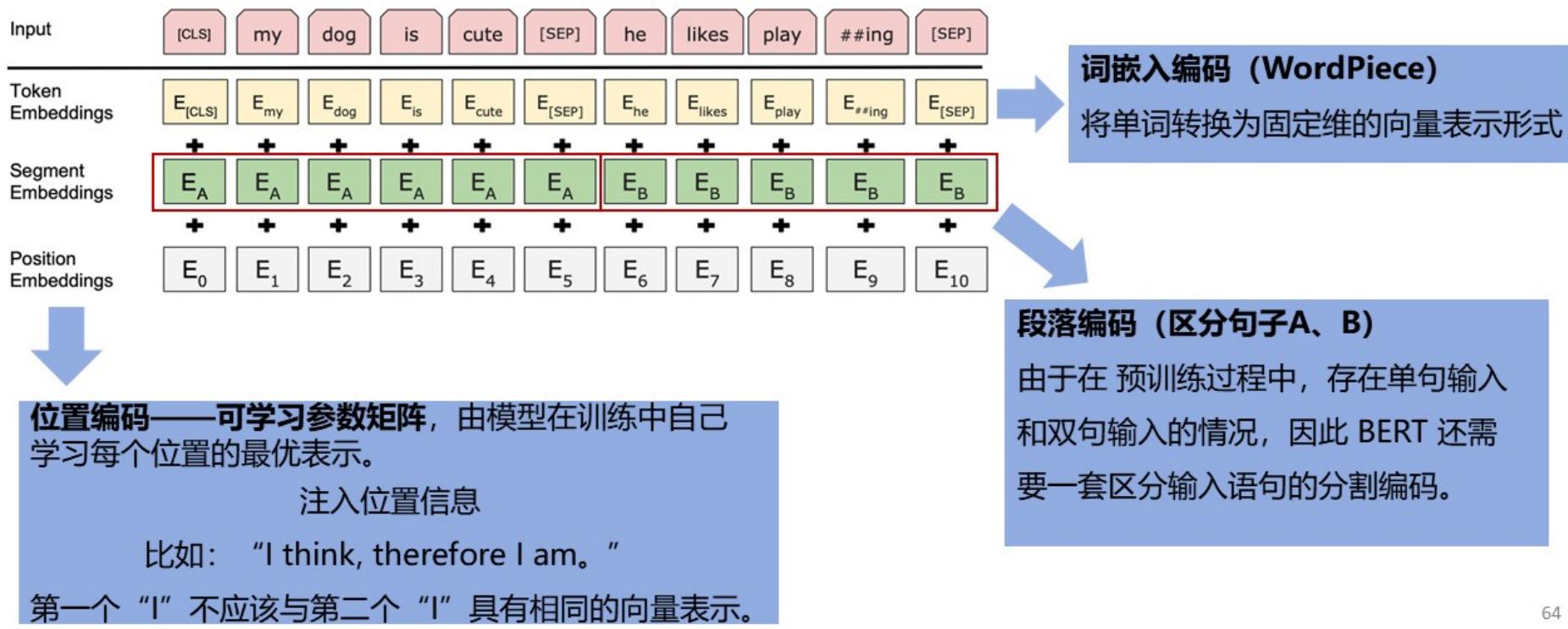
NSP是一个二元分类任务，目标是判断两个句子在原始文本中是否是连续的上下文关系，让模型学会判断两个句子的连贯性



其中 [SEP] 标签表示分隔符。 [CLS] 表示标签用于类别预测，  
IsNext为 1，表示输入为连续句对； IsNext结果为 0，表示输入为随机句对。

# 预训练语言模型：BERT

## ➤ BERT 之输入表示



# 预训练语言模型：BERT

## ➤ BERT 分词：WordPiece

### ➤ 基本思想

- 与 BPE 类似，但引入概率建模（基于似然）
- 选择能最大化训练数据似然的合并方式
- 保留更语义一致的子词

### ➤ 工作过程

1. 从字符开始
2. 评估每个候选合并对对数据似然的提升
3. 选择提升最大的对进行合并
4. 重复直到词表大小上限

词：internationalization

假设我们当前词表里出现了以下频繁组合：

"inter" → "international", "internet"

"nation" → "international", "nationality"

"ization" → "organization", "realization"

于是模型逐步合并得到：

inter | national | ization

并在词表中保存子词：

[ "inter", "#national", "#ization" ]

注：## 表示该子词不是词首

**优点：**低频词拆分更合理；对未知词（如新名词）有鲁棒性

**缺点：**依赖大规模语料；基于空格拆分

## 预训练语言模型：T5

### ➤ T5: Text-to-Text Transfer Transformer

BERT 善于理解文本，GPT 善于生成文本，  
但二者在任务形式上差异巨大：

模型	架构	任务形式	限制
GPT	Decoder-only	生成下一个词	只会写，不理解
BERT	Encoder-only	填空、分类	只会读，不能生成

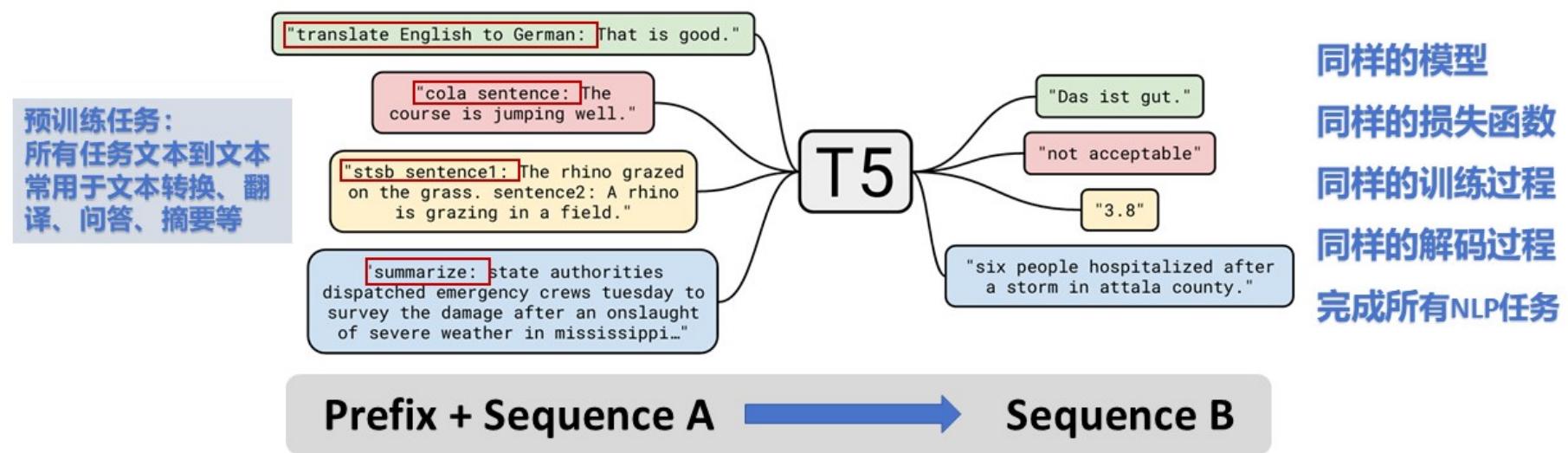
问题：

能不能用同一个模型，既能读也能写？

# 预训练语言模型：T5

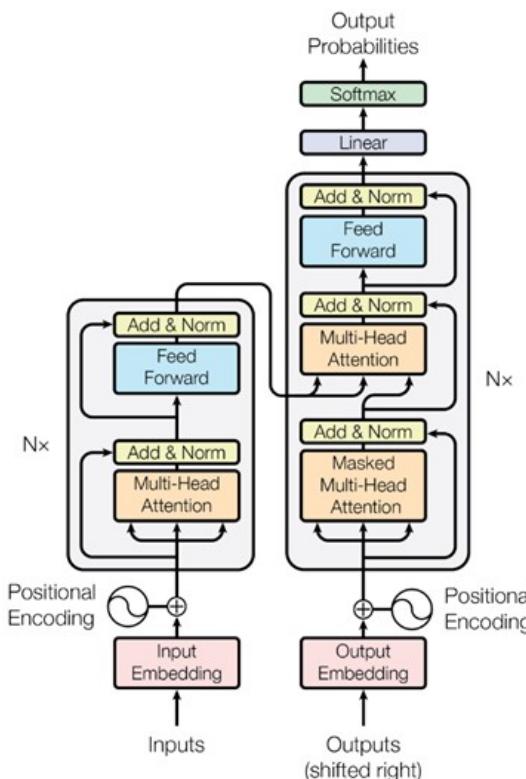
## ➤ T5: Text-to-Text Transfer Transformer (Google)

T5模型为所有下游任务提供了一个**统一的接口**。它为每个任务设计一个特定的**文本提示** (prompt)，将所有下游任务转换为**文本到文本 (Text-to-Text, 即序列到序列)** 的问题形式。



# 预训练语言模型：T5

## ➤ T5: Text-to-Text Transfer Transformer



- 基于 **Transformer Encoder–Decoder**
- Encoder: 读取输入语义
- Decoder: 按词生成输出文本

任务类型	输入	输出
翻译	“translate English to French: Hello”	“Bonjour”
摘要	“summarize: 今天我们学习了BERT模型...”	“BERT是一种理解式语言模型。”
分类	“classify sentiment: 这部电影太棒了！”	“positive”

## 预训练语言模型：T5

- **T5 预训练目标：**预测被遮挡的连续片段，而不是单词。
- T5 随机遮盖多个连续词片段（Span），并让模型重建这些内容。
- 遮盖的不是单个词，而是整段语义单元。
- 随机遮盖 15% 的 token，但以连续片段为单位

	原文本	Thank you for inviting me to your party last week
T5 (Span Corruption)	输入序列（随机替换15%的标记） 目标序列（重构替换的标记片段）	Thank you <X> me to your party <Y> week <X> for inviting <Y> last <Z>
Bert (MLM)	输入序列（随机替换15%的标记） 目标序列（输出原文本）	Thank you <M> <M> me to your party last week Thank you for inviting me to your party last week

BERT 做 “完形填空题”，T5 做 “补全作文题”。

## 预训练语言模型：T5

### ➤ T5 位置编码：Relative Position Bias (相对位置偏置)

在注意力分数中直接加一个可学习的偏置项：

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

其中  $B_{rel}(i, j)$  只依赖于位置差  $i - j$ ，而非绝对编号。

例如：

“今天” 在位置 3；

“明天” 在位置 4；

→  $i - j = -1$  (表示 “今天” 在 “明天” 前面 1 位)。

# 预训练语言模型：T5

## ➤ T5 位置编码：Relative Position Bias (相对位置偏置)

离散距离分桶，把词与词之间的相对距离  $i - j$  离散化为有限数量的“桶 (bucket)”，每个桶代表一类相对距离。通过把相对距离  $i - j$  映射到有限数量的桶（近细远粗），模型既能捕捉局部精确关系，又能保持远距离泛化能力。

稍远的位置就可以共用一个位置  
编码，距离越远，共用的范围  
越大达到指定范围再clip。

周围的位置 (0-7)  
分配一个独立的位置编码

$i - j$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$f(i - j)$	0	1	2	3	4	5	6	7	8	8	8	8	9	9	9	9
$i - j$	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	...
$f(i - j)$	10	10	10	10	10	10	10	11	11	11	11	11	11	11	11	...

# 预训练语言模型：T5

## ➤ T5 分词法：SentencePiece (Unigram LM)

### ➤ 基本思想

- 一种**语言无关**的分词算法
- 不依赖空格或人工规则
- 通过**概率模型** (Unigram Language Model) 选择最优子词集合

### ➤ 工作过程

1. 假设一组初始子词集合（庞大）
2. 计算每个子词在训练数据中的概率
3. 反复删除低概率子词，直到词表稳定
4. 训练后模型自动学习最优切分方式

例：我爱machine learning  
\_我\_爱\_machine\_learning  
假设词表中有以下候选：  
["\_我", "\_爱", "\_machine", "\_learning", "\_machine\_learning"]

切分	概率 (示意)
[_我, _爱, _machine, _learning]	1.4e-4
[_我, _爱, _machine_learning]	2.5e-4 <input checked="" type="checkbox"/>

**优点：**不依赖空格，适合多语言、多任务生成

**缺点：**子词边界不直观；有时会生成不符合语言习惯的分割

# 预训练语言模型：经典预训练任务

## ➤ 经典预训练任务

- 拓展补充：下图中是一些广泛使用的预训练任务。

Table 1: Loss Functions of Pre-training Tasks

经典自回归  
掩码语言模型  
序列到序列的MLM  
排列语言模型  
去噪干扰模型  
降噪干扰模型  
下一句预测/句序预测  
替换token检测

Task	Loss Function	Description
LM	$\mathcal{L}_{LM} = - \sum_{t=1}^T \log p(x_t   \mathbf{x}_{})$	$\mathbf{x}_{} = x_1, x_2, \dots, x_{t-1}$ .
MLM	$\mathcal{L}_{MLM} = - \sum_{\hat{x} \in m(\mathbf{x})} \log p(\hat{x}   \mathbf{x}_{\setminus m(\mathbf{x})})$	$m(\mathbf{x})$ and $\mathbf{x}_{\setminus m(\mathbf{x})}$ denote the masked words from $\mathbf{x}$ and the rest words respectively.
Seq2Seq MLM	$\mathcal{L}_{S2SMLM} = - \sum_{i=j}^j \log p(x_i   \mathbf{x}_{\setminus i:j}, \mathbf{x}_{i:t-1})$	$\mathbf{x}_{i:j}$ denotes an masked n-gram span from $i$ to $j$ in $\mathbf{x}$ .
PLM	$\mathcal{L}_{PLM} = - \sum_{t=1}^T \log p(z_t   \mathbf{z}_{})$	$\mathbf{z} = perm(\mathbf{x})$ is a permutation of $\mathbf{x}$ with random order.
DAE	$\mathcal{L}_{DAE} = - \sum_{t=1}^T \log p(x_t   \hat{\mathbf{x}}, \mathbf{x}_{})$	$\hat{\mathbf{x}}$ is randomly perturbed text from $\mathbf{x}$ .
DIM	$\mathcal{L}_{DIM} = s(\hat{\mathbf{x}}_{i:j}, \mathbf{x}_{i:j}) - \log \sum_{\hat{\mathbf{x}}_{i:j} \in \mathcal{N}} s(\hat{\mathbf{x}}_{i:j}, \hat{\mathbf{x}}_{i:j})$	$\mathbf{x}_{i:j}$ denotes an n-gram span from $i$ to $j$ in $\mathbf{x}$ , $\hat{\mathbf{x}}_{i:j}$ denotes a sentence masked at position $i$ to $j$ , and $\hat{\mathbf{x}}_{i:j}$ denotes a randomly-sampled negative n-gram from corpus.
NSP/SOP	$\mathcal{L}_{NSP/SOP} = - \log p(t   \mathbf{x}, \mathbf{y})$	$t = 1$ if $\mathbf{x}$ and $\mathbf{y}$ are continuous segments from corpus.
RTD	$\mathcal{L}_{RTD} = - \sum_{t=1}^T \log p(y_t   \hat{\mathbf{x}})$	$y_t = \mathbf{1}(\hat{x}_t = x_t)$ , $\hat{\mathbf{x}}$ is corrupted from $\mathbf{x}$ .

填空题 4分

请选择合适的答案填空。Transformer的结构是 [填空1]；GPT系列的结构是 [填空2]；Bert的结构是 [填空3]；T5的结构是 [填空4]。

- A. Encoder-Decoder
- B. Encoder
- C. Decoder
- D. 其他

# 预训练语言模型

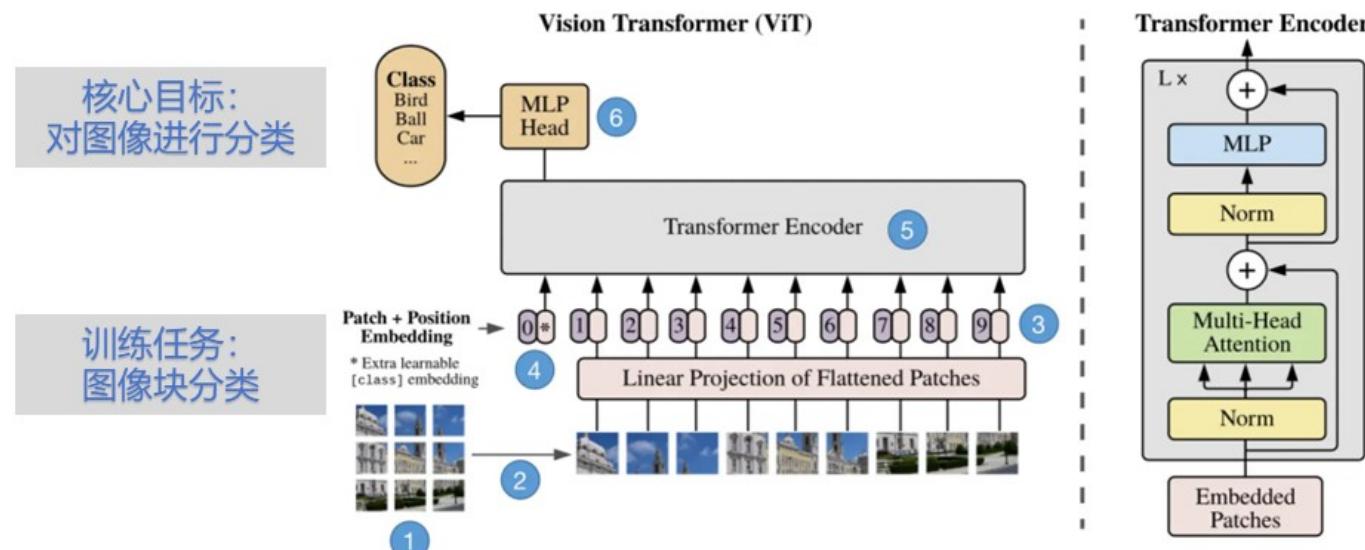
## ➤ 对比：

维度	GPT-1 (2018)	BERT (2018)	T5 (2019)	GPT-2 (2019)	GPT-3 (2020)
研发机构	OpenAI	Google	Google	OpenAI	OpenAI
架构	单向 Transformer 解码器	双向 Transformer 编码器	编码器-解码器 Transformer	单向 Transformer 解码器	单向 Transformer 解码器
训练目标	自回归语言模型 (预测下一个词)	MLM+NSP	多任务统一框架 (文本到文本)	自回归语言模 型 (更大规模)	自回归语言模 型 (超大规模)
参数量	1.17亿	1.1亿 (Base) 3.4亿 (Large)	6000万 (Small) 2.2亿 (Base) 110亿 (Large)	15亿	1750亿
关键创新	预训练+任务微调	引入双向上下 文表征 + 掩码 语言模型	统一文本任务为 “文本到文本”	Zero-shot 零样本	in-context learning 上下文学习

# 预训练视觉模型

## ➤ Vision Transformer (ViT): “图像的BERT”

ViT是2020年Google团队提出的将Transformer应用在图像分类的模型。



Dosovitskiy A. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arXiv preprint arXiv:2010.11929, 2020.

## 预训练视觉模型

### ➤ Vision Transformer (ViT): “图像的BERT”

使用大型标注数据集（如ImageNet），输入图像，模型输出一个类别标签（如“猫”、“狗”）。通过标准的交叉熵损失进行训练，目标是让预测的标签与真实标签一致。

1. 将图片转换成patches序列
2. 将patches铺平
3. 添加position embedding
4. 添加class token
5. 输入Transformer Encoder
6. 分类

## 预训练视觉模型

### ➤ Vision Transformer (ViT): “图像的BERT”

ViT 将图像视为“视觉序列”，在大规模图像数据上通过自监督或有监督预训练，获得可迁移的视觉表示。

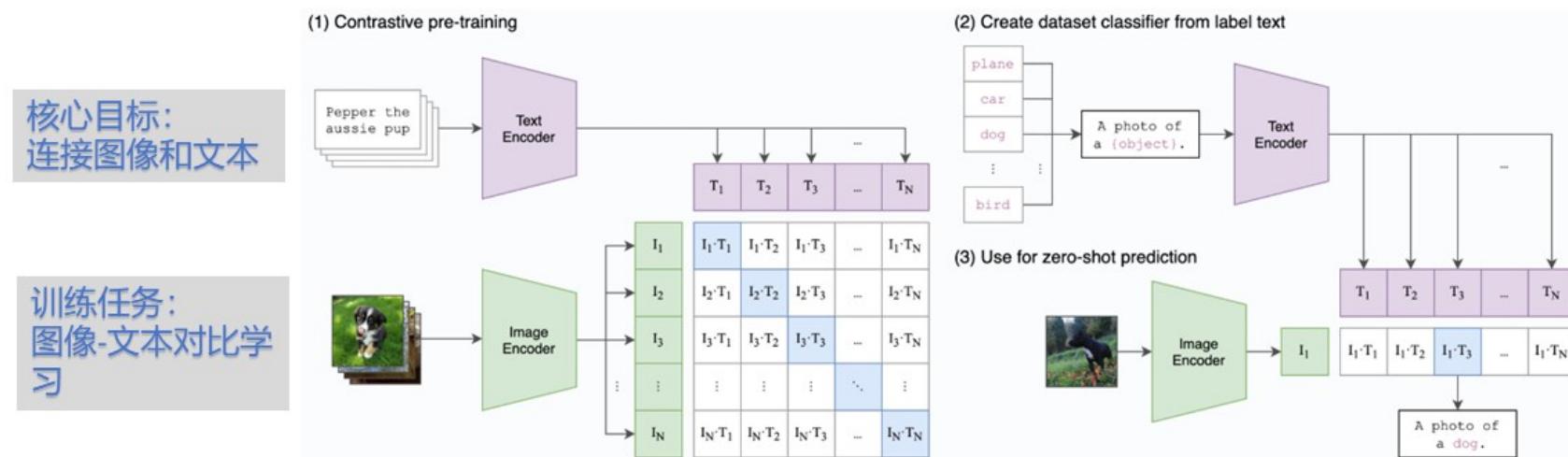
范式	任务描述	目标函数
<b>有监督分类式</b> (原版 ViT)	在 ImageNet 等标注数据上分类	交叉熵损失
<b>自监督重建式</b> (MAE)	随机遮盖 patch, 预测缺失区域	重建损失 (MSE / cross-entropy)

ViT 的核心是 通过海量图像预训练，学习视觉世界的通用分布。

# 预训练多模态模型

## ➤ CLIP：图文对比学习的里程碑——从单模态到多模态

CLIP (Contrastive Language-Image Pre-training) 是 OpenAI 于 2021 年提出的多模态模型，通过对比学习的方式建立图像和文本之间的关联。



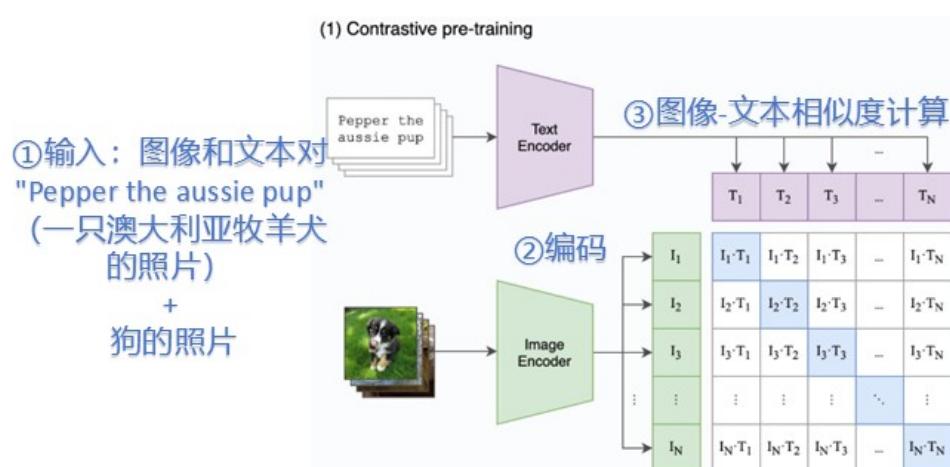
Radford A, Kim J W, Hallacy C, et al. Learning transferable visual models from natural language supervision[C]//International conference on machine learning. PMLR, 2021: 8748-8763.

# 预训练多模态模型

## ➤ CLIP: 图文对比学习的里程碑——从单模态到多模态

CLIP的两个核心组件：

- ✓ **图像编码器** (Image Encoder)：将图像转换为特征向量（Vision Transformer 或 ResNet）。
- ✓ **文本编码器** (Text Encoder)：将文本描述转换为特征向量（Transformer）。



```
# image_encoder - ResNet or Vision Transformer
# text_encoder - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l] - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T) #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

# 预训练多模态模型

## ➤ CLIP：图文对比学习的里程碑——从单模态到多模态

CLIP的对比学习：

**核心目标：**

- 让匹配的图文对在嵌入空间中相似度高，不匹配的对相似度低。实现语义对齐 (semantic alignment)
- 双向对比学习：
  - Image→Text
  - Text→Timage
- 损失函数 (InfoNCE 对比损失) InfoNCE = Cross-Entropy with softmax over similarity scores

$$L = \frac{1}{2N} \sum_{i=1}^N \left[ -\log \frac{\exp(\text{sim}(v_i, t_i)/\tau)}{\sum_{j=1}^N \exp(\text{sim}(v_i, t_j)/\tau)} - \log \frac{\exp(\text{sim}(t_i, v_i)/\tau)}{\sum_{j=1}^N \exp(\text{sim}(t_i, v_j)/\tau)} \right]$$

Image→Text    Text→Timage

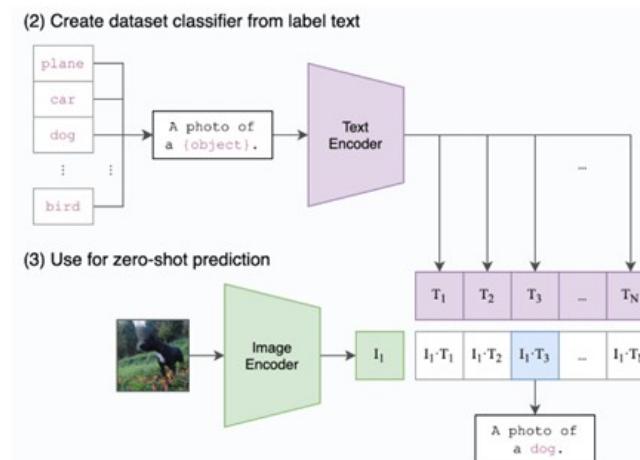
其中  $\text{sim}(v, t)$  为 cosine 相似度， $\tau$  为温度参数。

# 预训练多模态模型

## ➤ CLIP：图文对比学习的里程碑——从单模态到多模态

CLIP用于图像识别：

- 待识别的图片经过Image Encoder映射为特征向量；
- 同时将数据集中所有的类别文本转为“*A photo of a {object}.*”形式；
- 将这些新的文本经过Text Encoder映射到与图片一样的特征空间；
- 最后，找到与图像向量最近的文本向量。



# Lecture Plan

- 预训练模型的发展历程
- 预训练模型
- 微调技术
- 强化学习



83

# 微调技术

## ➤ 什么是微调

模型在与目标任务或领域相关的更具体、更小的数据集上进一步训练。让模型学习任务的**具体特征**，并适应任务的特殊要求。



84

# 预训练+微调 · 技术范式

## ➤ 预训练+微调范式

- 预训练+微调示例



85

## 传统微调技术：全参数微调

### ➤ 全参数微调

全参数微调（Full Fine-tuning）是指在预训练模型的基础上，使用下游任务的数据集，更新模型中的**所有参数**，以使模型适应特定任务。

微调模型与原始预训练模型的**大小相同**

模型变得越来越大



为每个下游任务独立存储和部署微调模型  
变得非常昂贵



**损失多样性 灾难性遗忘**

86

## 传统微调技术：全参数微调

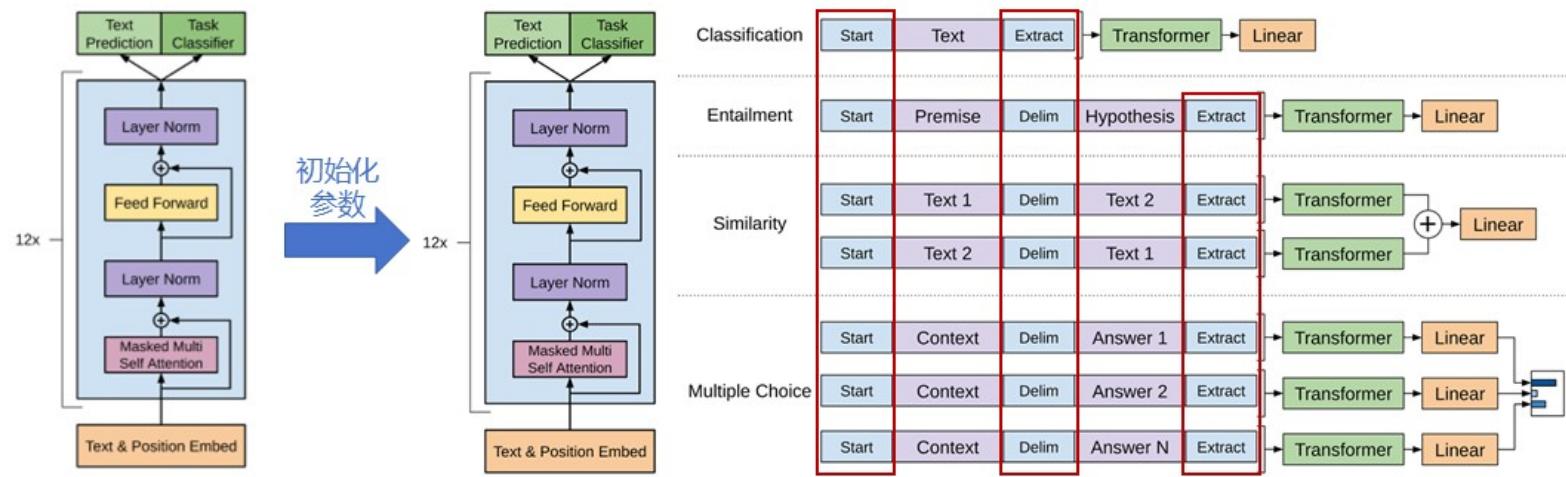
- 逐层冻结策略 (Layer-wise freezing)

87

# 预训练语言模型：GPT1

## ➤ GPT-1 Fine-tuning:

在 GPT-1 模型的下游任务中，需要根据 GPT-1 的网络结构，对下游任务做适当的修改，针对不同的下游任务，需要对输入进行转换：



在已初始化的模型上，使用特定下游任务的数据进行训练

Task-specific  
Fine-tuning

# 预训练语言模型：GPT1

## ➤ Task-specific Fine-tuning

通常微调的任务目标取决于**具体的下游任务**的性质。

以下列举了几种NLP有关的下游任务：

- **Single-text Classification (单句分类) :**
- 常见的单句分类任务有短文本分类、长文本分类、意图识别、情感分析、关系抽取等。

- 短/长文本分类：直接对句子进行归类，例如新闻归类、主题分类、场景识别等；
- 意图识别：根据给定的问句判断其意图，常用于检索式问答、多轮对话、知识图谱问答等；
- 情感分析：对评论类型的文本进行情感取向分类或打分；
- 关系抽取：给定两个实体及对应的一个描述类句子，判断这两个实体的关系类型；

# 预训练语言模型：GPT1

## ➤ Task-specific Fine-tuning

- **Sentence-pair Classification (句子匹配/成对分类) :**
  - 语义推理/蕴含：判断两个句子是否存在推理关系，例如entailment、contradiction, neutral三种推理关系；
  - 文本匹配与检索：输入一个文本，并从数据库中检索与之高相似度匹配的其他句子

- **Span Text Prediction (区间预测) :**

- 常见的任务类型有抽取式阅读理解、实体抽取、抽取式摘要等。
  - 抽取式阅读理解：给定query和passage，寻找passage中的一个文本区间作为答案；
  - 实体抽取：对一段文本中寻找所有可能的实体；
  - 抽取式摘要：给定一个长文本段落，寻找一个或多个区间作为该段落的摘要；

# 预训练语言模型：GPT1

## ➤ Task-specific Fine-tuning

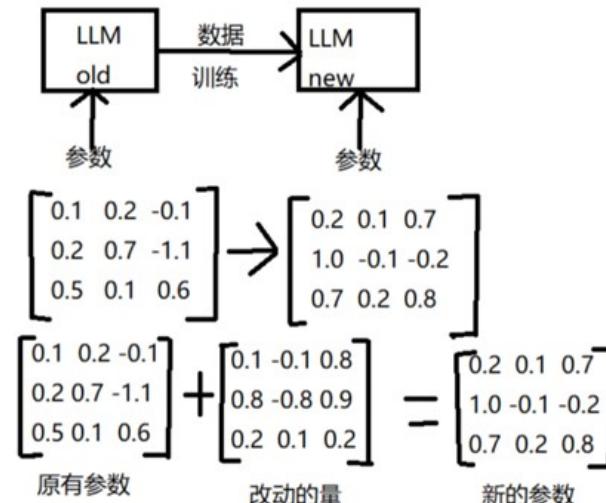
- **Single-token Classification (字符分类) :**
  - 序列标注：对给定的文本每个token进行标注，通常有词性标注、槽位填充、句法分析、实体识别等；
  - 完形填空：与MLM一致，预测给定文本中空位处可能的词
  - 拼写检测：对给定的文本中寻找在语法或语义上的错误拼写，并进行纠正；
- **Text Generation (文本生成) :**
  - 生成式摘要：在文本摘要中，通过生成方法获得摘要；
  - 机器翻译：给定原始语言的文本，来生成目标语言的翻译句子；
  - 问答：给定query，直接生成答案；

# 微调技术：参数高效微调

## ➤ 参数高效微调 (PEFT)

参数高效微调 (Parameter-Efficient Fine-Tuning, PEFT) 是一系列旨在以较少的计算资源和数据量，实现与全参数微调相近性能的技术。这类方法通常**冻结**预训练模型的**大部分参数**，只训练少量额外的参数。

- LoRA系列
- 提示微调技术
  - Prefix-Tuning
  - P-Tuning
  - Prompt-Tuning



92

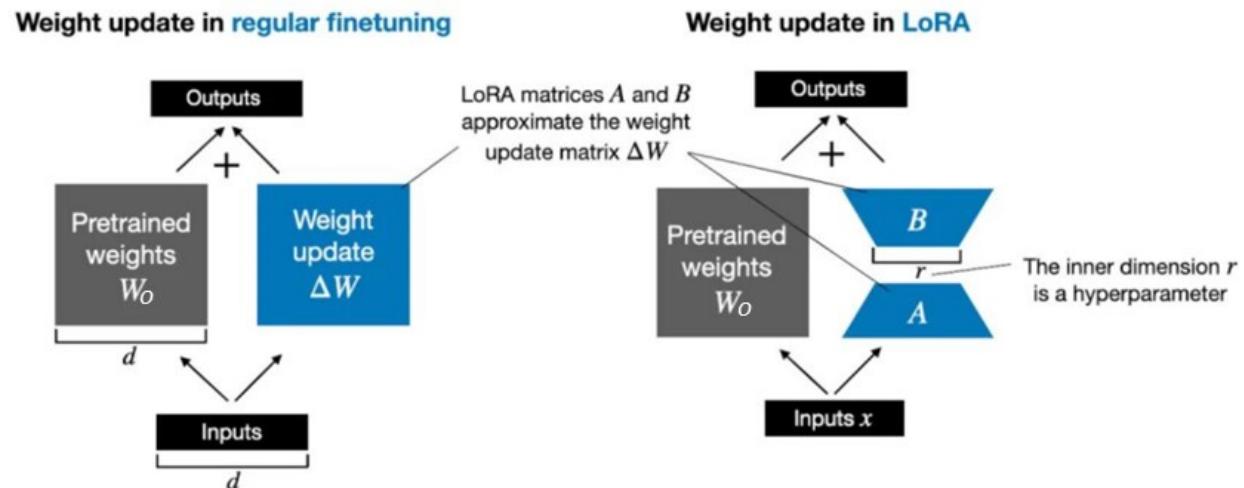
# 微调技术：LoRA

## ➤ 参数高效微调 - LoRA系列 - LoRA

- 低秩适配（LoRA）的LoRA的核心是**冻结预训练模型原始参数**，通过**低秩分解**引入可训练参数，参数量少但性能接近全参数微调。

$$W = W_o + \Delta W$$

$\Delta W$ : 需要更新的参数  
 $W_o$ : 预训练模型初始化的参数

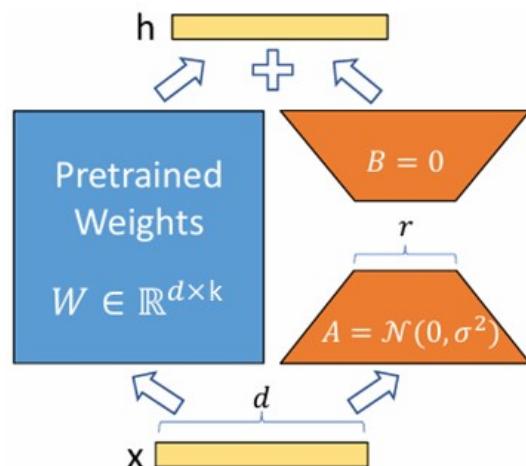


93

# 微调技术：LoRA

## ➤ 参数高效微调 - LoRA系列 - LoRA

- 具体来说，对于一个预训练好的权重矩阵  $W_0 \in \mathbb{R}^{d \times k}$



- LoRA 引入两个**低秩矩阵**

$$A \in \mathbb{R}^{d \times r} \text{ 和 } B \in \mathbb{R}^{r \times k}$$

- 其中  $r \ll \min(d, k)$  是**秩**。
- 在微调过程中，只优化  $A$  和  $B$ ，而  $W_0$  保持不变。

**更新后的权重矩阵为：**

$$W = W_0 + BA$$

# 微调技术：AdaLoRA

## ➤ 参数高效微调 - LoRA系列 - AdaLoRA

- LoRA需要预先指定每个增量矩阵的本征秩  $r$  相同，忽略了在微调预训练模型时，权重矩阵的重要性在不同模块和层之间存在显著差异。
- **AdaLoRA**，是对LoRA的一种改进，它根据重要性评分**动态分配**参数预算给权重矩阵。具体做法如下：
  - 1、调整增量矩分配。
  - 2、以奇异值分解的形式对增量更新进行参数化，并根据重要性指标裁剪掉不重要的奇异值，同时保留奇异向量。
  - 3、在训练损失中添加了额外的惩罚项。

Algorithm 1 AdaLoRA

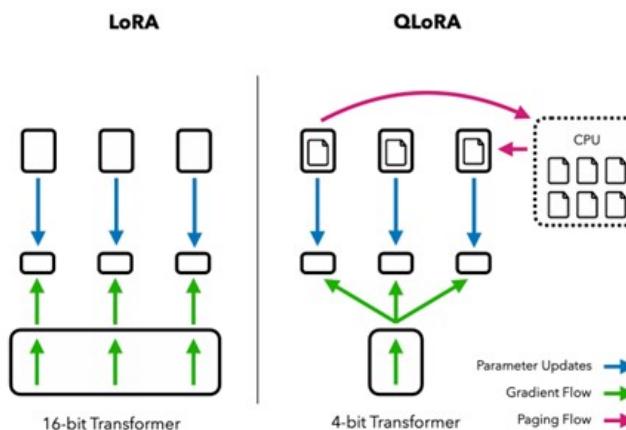
```
1: Input: Dataset  $\mathcal{D}$ ; total iterations  $T$ ; budget schedule  $\{b^{(t)}\}_{t=0}^T$ ; hyperparameters  $\eta, \gamma, \beta_1, \beta_2$ .  
2: for  $t = 1, \dots, T$  do  
3:   Sample a mini-batch from  $\mathcal{D}$  and compute the gradient  $\nabla \mathcal{L}(\mathcal{P}, \mathcal{E}, \mathcal{Q})$ ;  
4:   Compute the sensitivity  $I^{(t)}$  in (8) for every parameter in  $\{\mathcal{P}, \mathcal{E}, \mathcal{Q}\}$ ;  
5:   Update  $\bar{I}^{(t)}$  as (9) and  $\bar{U}^{(t)}$  as (10) for every parameter in  $\{\mathcal{P}, \mathcal{E}, \mathcal{Q}\}$ ;  
6:   Compute  $S_{k,i}^{(t)}$  by (7), for  $k = 1, \dots, n$  and  $i = 1, \dots, r$  ;  
7:   Update  $P_k^{(t+1)} = P_k^{(t)} - \eta \nabla_{P_k} \mathcal{L}(\mathcal{P}, \mathcal{E}, \mathcal{Q})$  and  $Q_k^{(t+1)} = Q_k^{(t)} - \eta \nabla_{Q_k} \mathcal{L}(\mathcal{P}, \mathcal{E}, \mathcal{Q})$ ;  
8:   Update  $\Lambda_k^{(t+1)} = \mathcal{T}(\Lambda_k^{(t)} - \eta \nabla_{\Lambda_k} \mathcal{L}(\mathcal{P}, \mathcal{E}, \mathcal{Q}), S_k^{(t)})$  given the budget  $b^{(t)}$ .  
9: end for  
10: Output: The fine-tuned parameters  $\{\mathcal{P}^{(T)}, \mathcal{E}^{(T)}, \mathcal{Q}^{(T)}\}$ .
```

95

# 微调技术：QLoRA

## ➤ 参数高效微调 - LoRA系列 - QLoRA

- QLoRA将4-bit 量化与 LoRA 相结合，以进一步降低显存占用。
- QLoRA提出了两种技术实现高保真**4 bit**微调——**4 bit NormalFloat(NF4) 量化和双量化**。此外，还引入了**分页优化器**，用来在显存过高时用一部分内存代替显存。



96

# 微调技术：提示微调技术

## ➤ 提示微调技术

- 提示微调（Prompt Tuning）是一种通过设计合适的提示（Prompt）来引导预训练模型完成下游任务的技术。
- 与全参数微调和 LoRA 不同，提示微调通常不直接修改预训练模型的参数（注意不是完全不修改参数），而是通过优化提示相关的向量来调整模型的行为。

人工设计 → 可学习提示

$$\mathbf{h}_{prompt} = [\mathbf{P}_1; \mathbf{P}_2; \dots; \mathbf{P}_k] \quad \mathbf{P}_i \in \mathbb{R}^d \quad \text{可学习的提示向量}$$

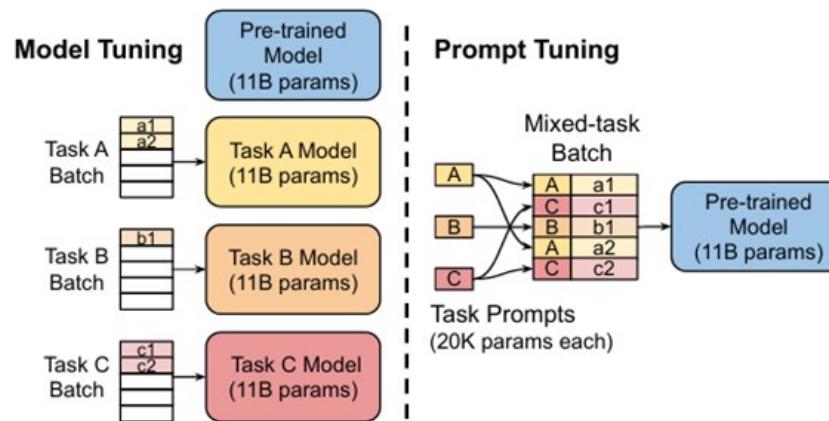
$\mathbf{h}_{prompt}$  表示提示向量， $P_i$  表示第  $i$  个可训练的提示向量， $k$  表示提示的长度， $d$  表示提示向量的维度。

97

# 微调技术: Prompt Tuning

## ➤ 提示微调技术-Prompt Tuning

- 给每个任务定义了自己的Prompt，然后拼接到数据上作为输入，但只在输入层加入prompt tokens。
- 通过反向传播更新参数来学习prompts，而不是人工设计prompts；同时冻结模型原始权重，只训练与prompts相关的嵌入向量。

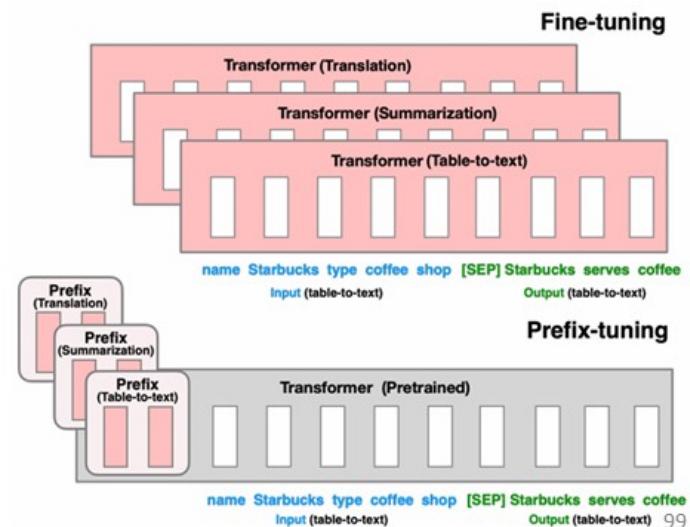


98

# 微调技术: Prefix Tuning

## ➤ 提示微调技术-Prefix Tuning

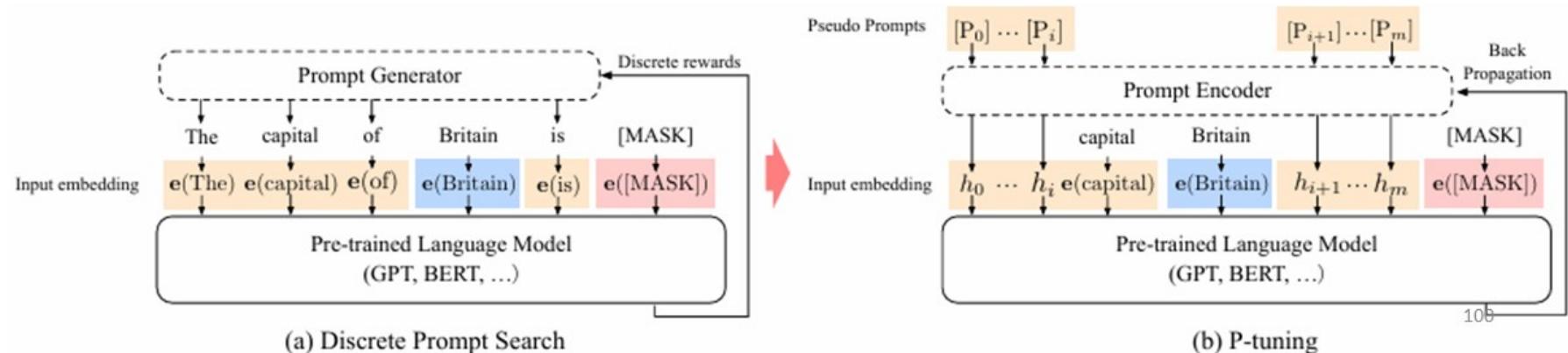
- Prefix Tuning的核心思想是在Transformer的每一层输入中，加入一组可训练的“前缀向量”，作为额外的上下文信息，参与注意力机制的计算，从而调整模型的输出。
- 只需训练前缀向量，不许调整模型原有参数
- 训练成本低



# 微调技术：P-Tuning

## ➤ 提示微调技术 - P-Tuning

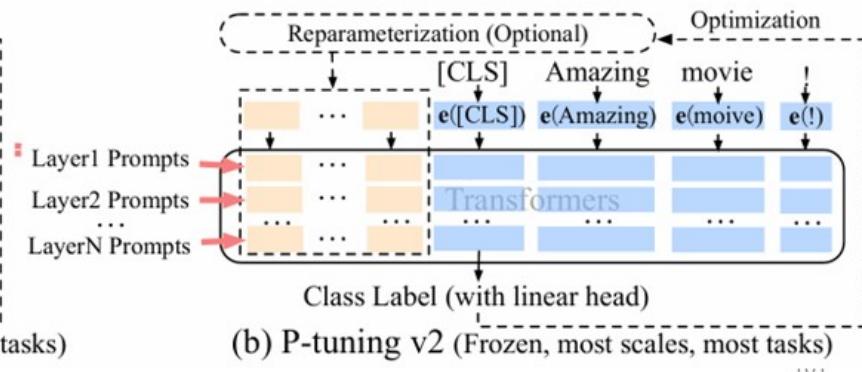
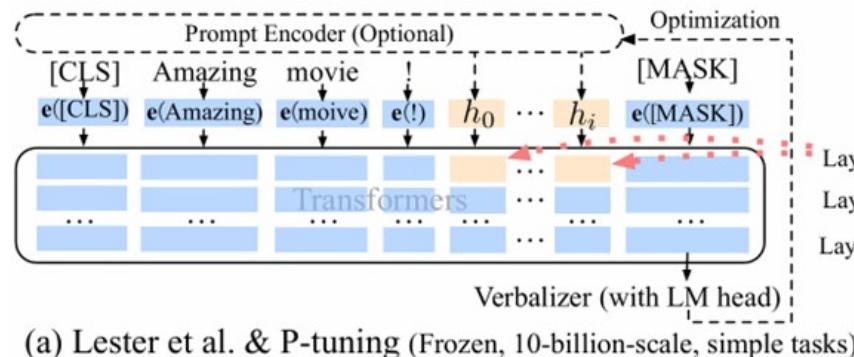
- 将Prompt转换为可以学习的Embedding层，并用MLP+LSTM的方式来对Prompt Embedding进行一层处理。
- 相比Prefix Tuning，P-Tuning加入的virtual token仅限于输入层，没有在每一层都加；另外，virtual token的位置也不一定是前缀，插入的位置是可选的。



# 微调技术：P-Tuning v2

## ➤ 提示微调技术 - P-Tuning v2

- 该方法在每一层都加入了Prompts tokens作为输入，而不是仅仅加在输入层，这带来两个方面的好处：
  - 更多可学习的参数，同时也足够参数高效。
  - 加入到更深层结构中的Prompt能给模型预测带来更直接的影响。



# 微调技术：领域自适应微调

## ➤ 领域自适应微调

领域自适应微调（Domain Adaptive Fine-Tuning）是指在**特定领域**的数据上对预训练模型进行微调，以使其更好地适应该领域的任务。这种方法在医疗、法律等专业领域尤其重要，因为这些领域的数据具有独特的特点和术语。

"请解释什么是高血压？"

"如何预防感冒？"

《柳叶刀》杂志最近发表了关于 COVID-19 疫苗有效性的研究，请总结其主要发现。

102

- 增加一个应用案例，比如使用huggingface的transformer库如何实现微调一个预训练模型

103

# Lecture Plan

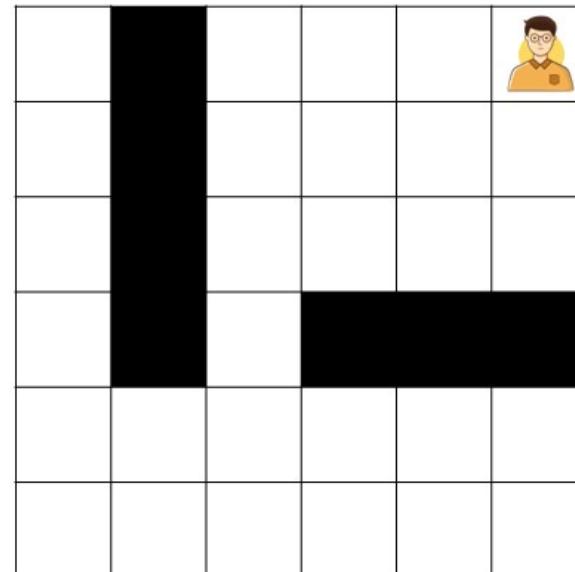
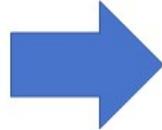
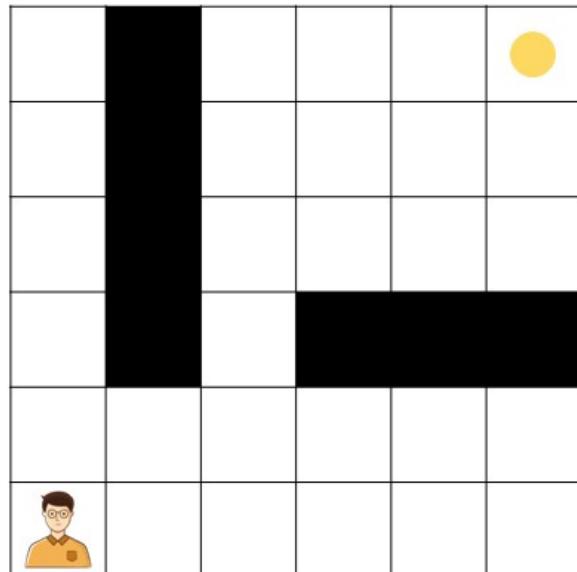
- 预训练语言模型的发展历程
- 预训练语言模型
- 微调技术
- 强化学习



104

## 吃豆人小游戏（豆子固定）

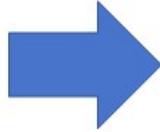
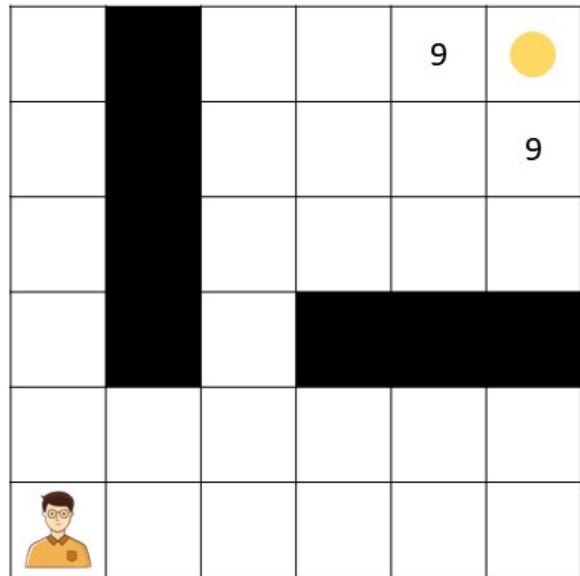
- Play A Game! 这是一个棋盘，棋盘上有一颗豆，你站在棋盘上可以自由移动，黑色的部分是障碍，吃到这颗豆就可以得十分。



换一个案例【不需要一步一步的推，可以就举几个引用场景，比如 alphago，自动驾驶，游戏智能体】，形象的解释为什么要强化学习，以及它是什么？

# 吃豆人小游戏（豆子固定）

看到9就知道很快  
能吃到豆子

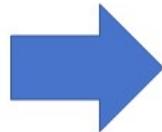
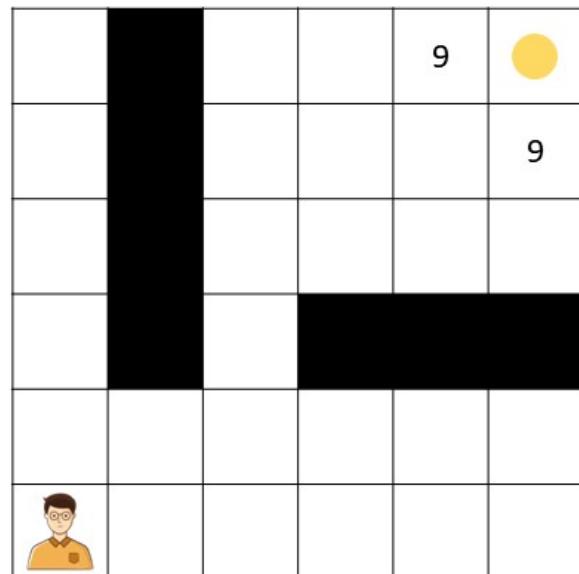


-3		7	8	9	9
-2		6	7	8	9
-1		5	6	7	8
0		4			
1	2	3	2	1	0
1	2	1	0	0	-1

现在无论在棋盘的  
哪个位置都可以顺  
着数字很快的吃到  
豆子

## 吃豆人小游戏（豆子随机）

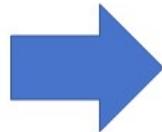
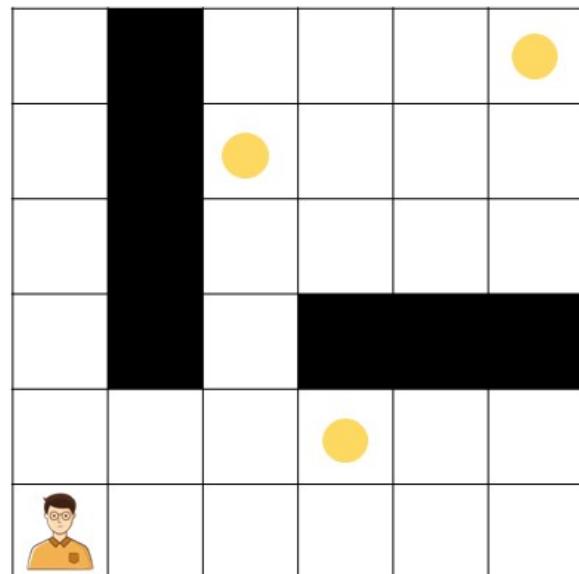
每吃完一次豆子，就会随机出现新的豆子，你就会擦掉棋盘上的数字，重复刚才的过程，当然最好的方法就是记住每一种情况，那就可以快速反应。



-3		7	8	9	9
-2		6	7	8	9
-1		5	6	7	8
0		4			
1	2	3	2	1	0
1	2	1	0	0	-1

# 吃豆人小游戏（随机出现多颗豆子且一段时间后消失）

棋盘上随机出现多颗豆子，且五步内未吃到豆子就会消失。依然可以标数字，但是现在的情况变复杂了，很难准确标出每一个位置的数字，只能不断的修改。

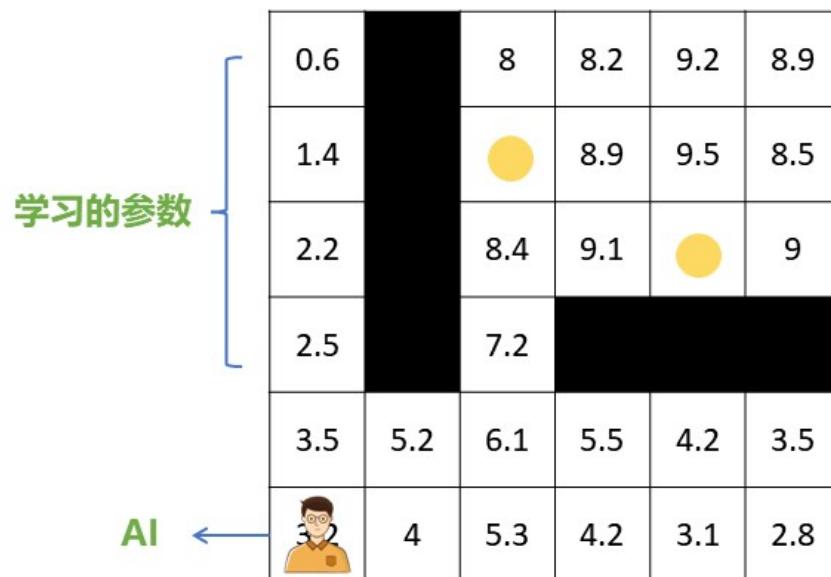


0.6		8	8.2	9.2	8.9
1.4		●	8.9	9.5	8.5
2.2		8.4	9.1	●	9
2.5		7.2			
3.5	5.2	6.1	5.5	4.2	3.5
3.2	4	5.3	4.2	3.1	2.8

如果一个位置总是能很快吃到豆子，那这个位置就会标上很大的数字。经过几千次游戏之后，针对各种情况的数字逐渐固定下来。

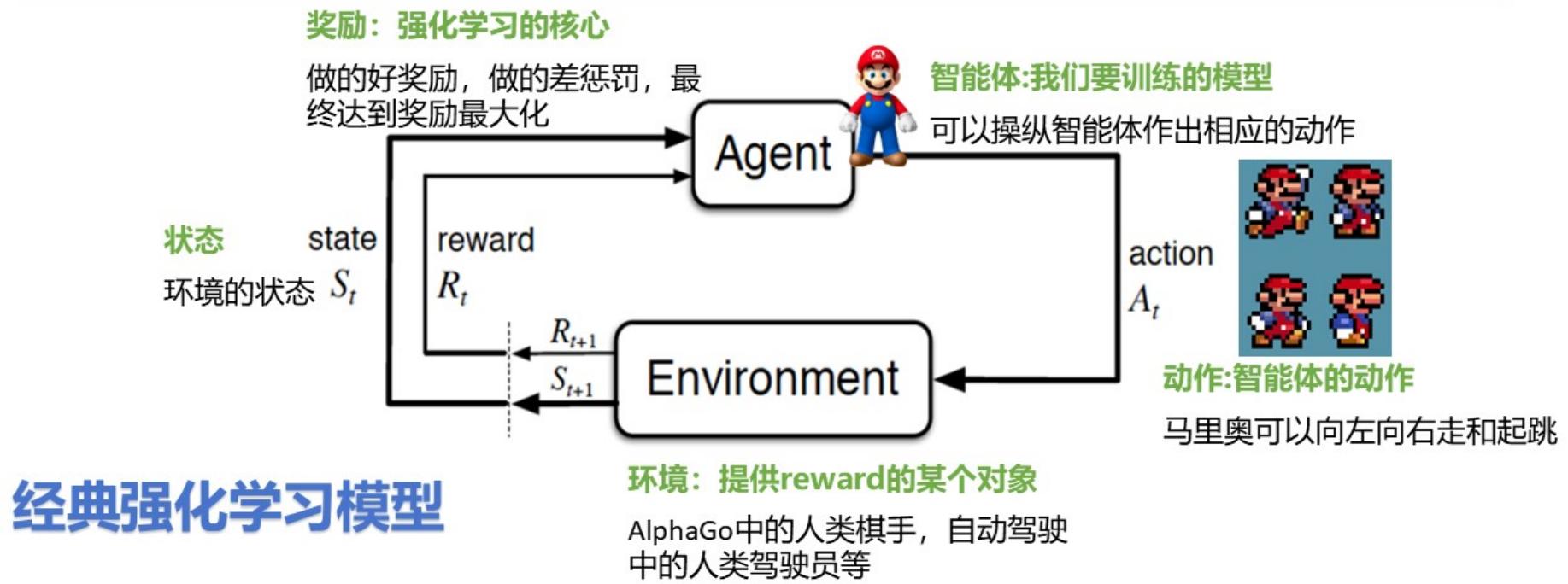
# 强化学习

强化学习在游戏、自动驾驶等领域非常常见。比如AlphaGo和Alphazero，训练思路就是设置奖励，让AI在学习中自己学习，追求奖励最大化。



# 强化学习：经典强化学习模型

强化学习是指基于智能体在复杂、不确定的环境中**最大化它能获得的奖励**，从而达到**自助决策**的目的。



# 强化学习：经典强化学习模型

强化学习的关键是什么？奖励函数还是什么？不同的强化学习算法遵循相同的范式，那他们主要的研究围绕那一块儿呢？

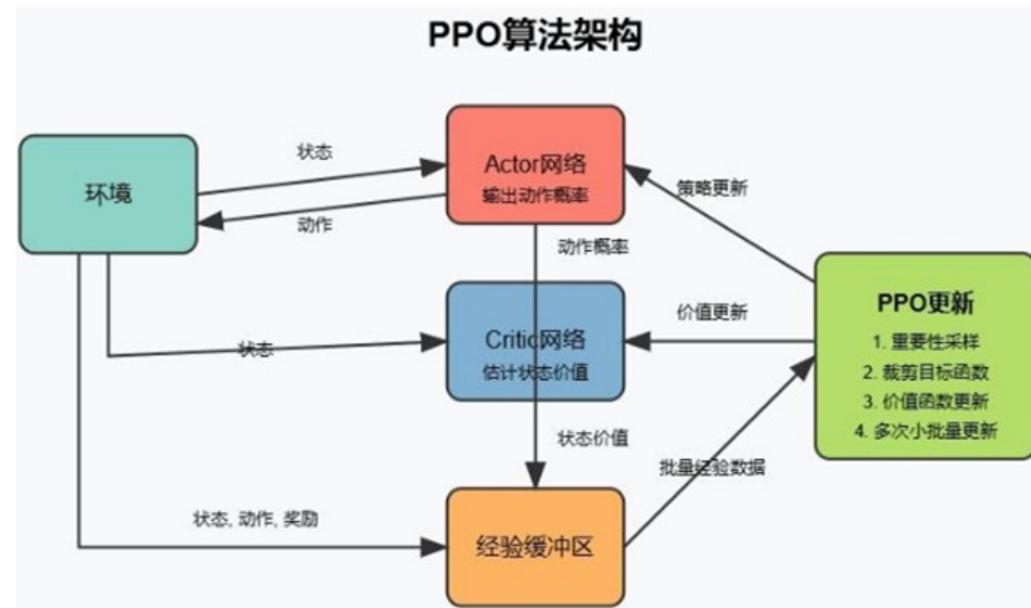
【基于前面的抽出一个伪代码】马尔可夫决策过程、策略学习与价值函数，Q-learning, DQN, Policy Gradient, 然后在围绕大模型RLHF中的PPO与DPO。

# 强化学习：PPO

## ➤ 优化算法-近端策略优化 (PPO)

对变化幅度进行约束

- PPO 是 OpenAI 提出的强化学习算法，属于基于**策略梯度** (Policy Gradient) 的强化学习方法，采用 Actor-Critic 结构，结合策略梯度和价值函数估计进行训练。



112

# 强化学习

## ➤ 优化算法-近端策略优化 (PPO)

- PPO 通过定义目标函数来控制策略更新的幅度。
- PPO 采用了信赖域策略优化 (Trust Region Policy Optimization, TRPO) 的思想，但用更简单的裁剪方式来限制策略更新。

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[ \min(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t) \right]$$

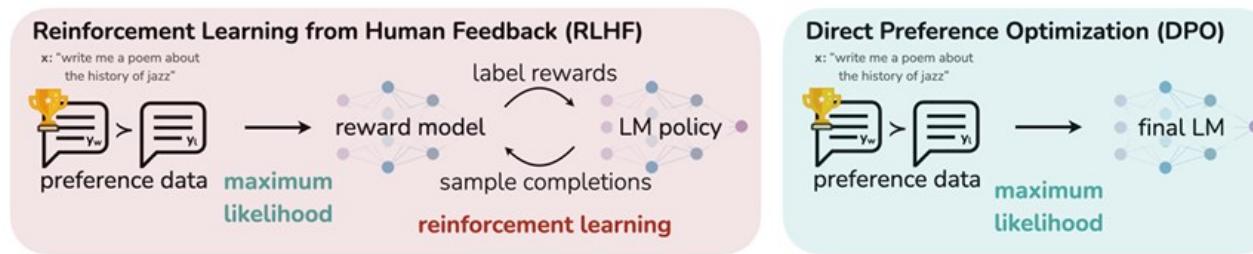
- 通过裁剪比率  $r_t(\theta)$ ，PPO 避免了策略过度更新，保证了训练的稳定性。

113

# 强化学习

## ➤ 优化算法-直接偏好优化 (DPO)

- DPO专注于基于用户偏好的学习，而非传统的奖励信号优化。
- 在强化学习中，奖励信号是关键。但在许多任务（如对话生成、推荐系统、AI生成内容）中，奖励函数难以明确定义。因此，DPO 提出了一种新的思路：**直接优化用户的偏好**，而非构造奖励函数。



114

## 强化学习

### ➤ 优化算法-直接偏好优化 (DPO)

- DPO优化目标：

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[ \log \sigma \left( \beta \log \frac{\pi_\theta(y_w | x)}{\pi_{\text{ref}}(y_w | x)} - \beta \log \frac{\pi_\theta(y_l | x)}{\pi_{\text{ref}}(y_l | x)} \right) \right]$$

- 该目标增加了对偏好数据 $y_w$ 的可能性，并减少了非偏好数据 $y_l$ 的可能性。

115

## 强化学习

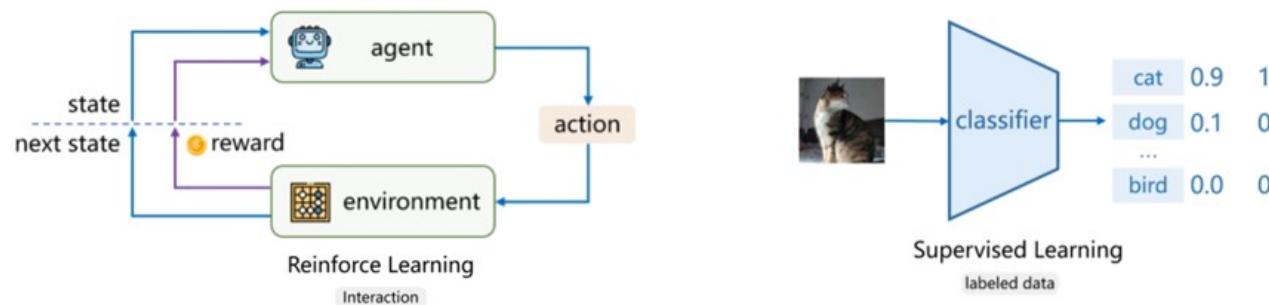
### ➤ 优化算法-直接偏好优化 (DPO)

- 相比PPO，DPO优势在于不需要训练奖励模型，可以直接使用偏好数据。
- 但同时，其对偏好数据的质量要求更高，数量要求也更多。

116

# 强化学习与监督学习的区别

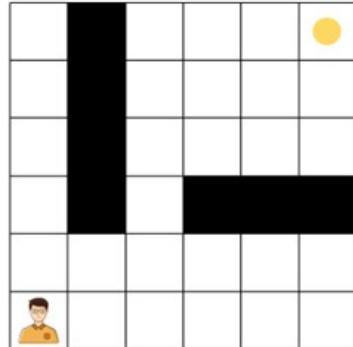
	强化学习	监督学习
	无标签	有标签
目标	最大化累计的奖励期望	最小化给定的损失函数
最优策略	$\arg \max E\{[\text{奖励函数(状态, 动作)}]\}$	$\arg \min E\{[\text{损失函数(标签, 模型(特征))}]\}$
反馈机制	延迟反馈	及时反馈
输入	当前行为会影响下一次决策的输入	输入是独立分布



# 强化学习

## ➤ 问题：奖励函数从何而来？

- 思考



吃豆人小游戏



AlphaGo

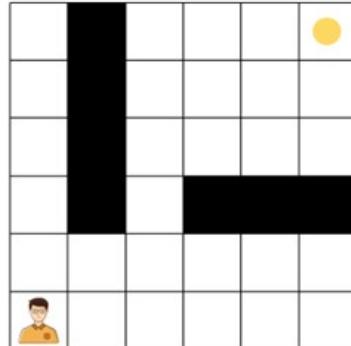


创意绘画AI

# 强化学习

## ➤ 问题：奖励函数从何而来？

- 思考



吃豆人小游戏



AlphaGo



创意绘画AI

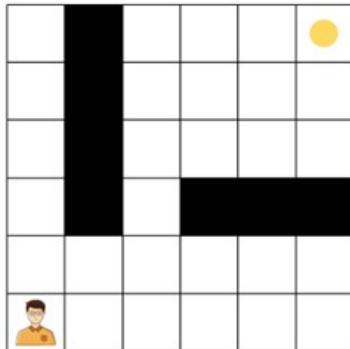
通过以上三个问题，大家会发现在模拟环境（如游戏）中，奖励是清晰、可量化的（得分、输赢）。  
但在许多现实任务中，“好”的标准是模糊、复杂且多维度的。

119

# 强化学习

## ➤ 问题：奖励函数从何而来？

- 思考



吃豆人小游戏



AlphaGo



创意绘画AI

既然定义“好”如此困难，那我们不直接定义奖励，而是让人类来“评价”什么样的行为是“好”的。

将人类的模糊偏好和价值观，转化为AI模型可以理解和优化的信号。

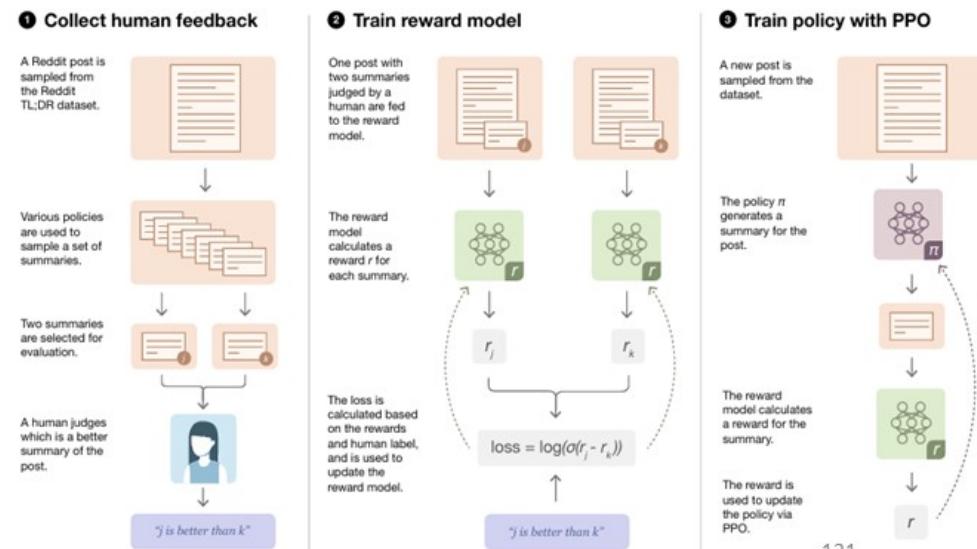
**RLHF**

# 强化学习：RLHF

## ➤ RLHF：人类反馈强化学习

- RLHF就是**基于人类反馈**对语言模型进行强化学习
- RLHF的训练过程可以分解为三个核心步骤：

- 1、多种策略产生样本并收集人类反馈
- 2、训练奖励模型
- 3、训练强化学习策略

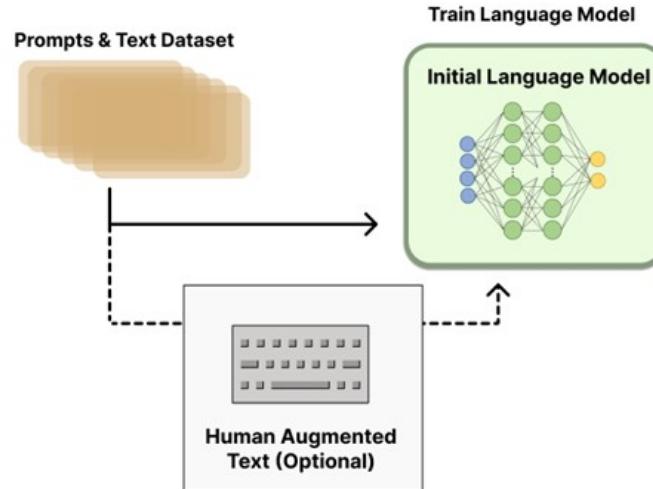


121

# 强化学习：RLHF-Step1

## ➤ RLHF：人类反馈强化学习

- Step 1：预训练语言模型+有标签数据微调（可选）
- 首先需要一个预训练语言模型，通过**大量的语料**去训练出**基础模型**。还有一个可选的Human Augmented Text。这里说直白点就是招人给问题写示范回答，然后给语言模型上学习。
- 但实际想要用人工去撰写答案的方式来训练模型，那成本是不可想象的，所以需要引入强化学习。

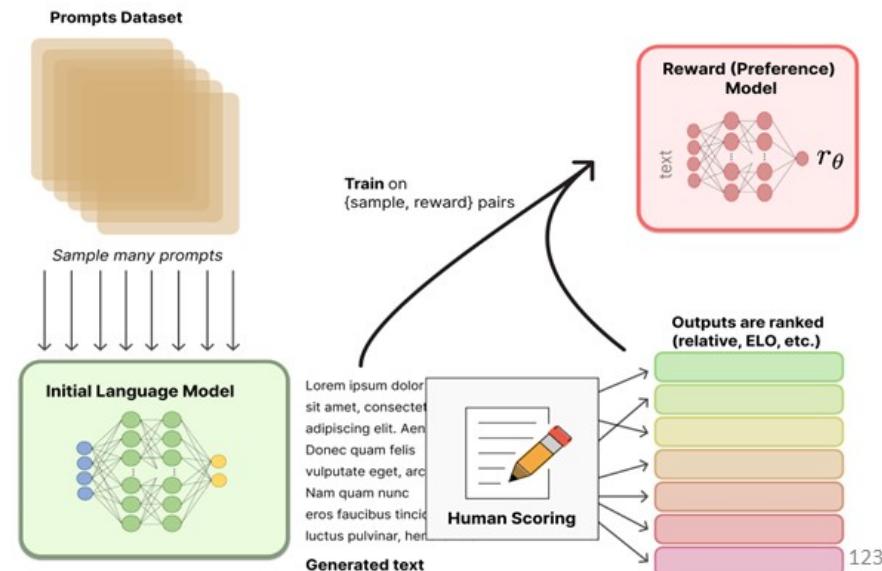


122

# 强化学习：RLHF-Step2

## ➤ RLHF：人类反馈强化学习

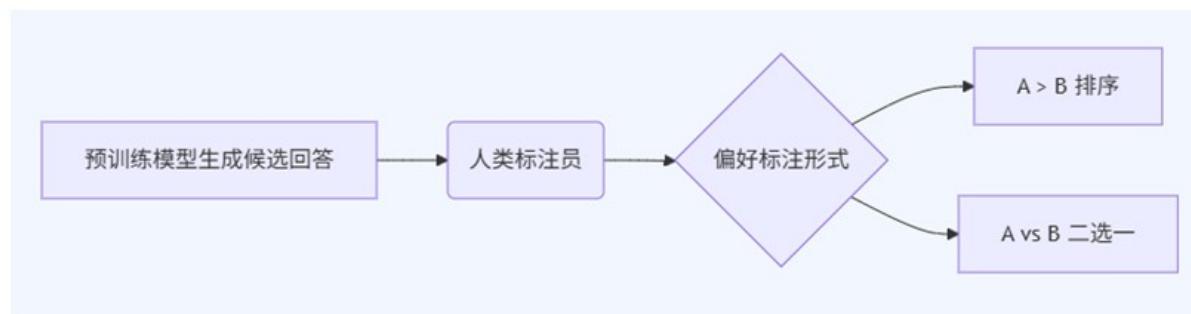
- Step 2: 训练奖励模型
  - RM 的训练是 RLHF 区别于旧范式的开端。这一模型接收一系列文本并返回一个**标量奖励**，数值上对应**人的偏好**。
- 常见的两种奖励模型：
- Outcome Reward Model (ORM):  
ORM通过对生成的最终答案进行评分，来评估模型的表现。
  - Process Reward Model (PRM):  
PRM不仅仅关注最终答案，还对模型的推理过程进行评分。



# 强化学习-Step2.1

## ➤ Reward Model训练过程

- 输入数据：人类偏好数据
  1. 收集**候选输出**：使用预训练的语言模型生成多个候选输出。
  2. 人类**打分或偏好标注**：人类对多个候选输出进行排序或选择偏好的选项。例如，对于一组生成结果 A 和 B，标注  $A > B$ ，表示 A 更符合偏好。
  3. **数据格式**：偏好数据通常以  $(A, B)$  的形式表示。



124

## 强化学习-Step2.2

### ➤ Reward Model训练过程

- 训练目标
- Reward Model 的**目标**是学习一个奖励函数  $r(x,y)$ ，其中 $x$ 是提示， $y$ 是响应,输出一个标量奖励值  $r$ 。
- 训练时，Reward Model需要确保：如果人类认为  $y_1 > y_2$  (响应1优于响应2) ，则  $r(x,y_1) > r(x,y_2)$



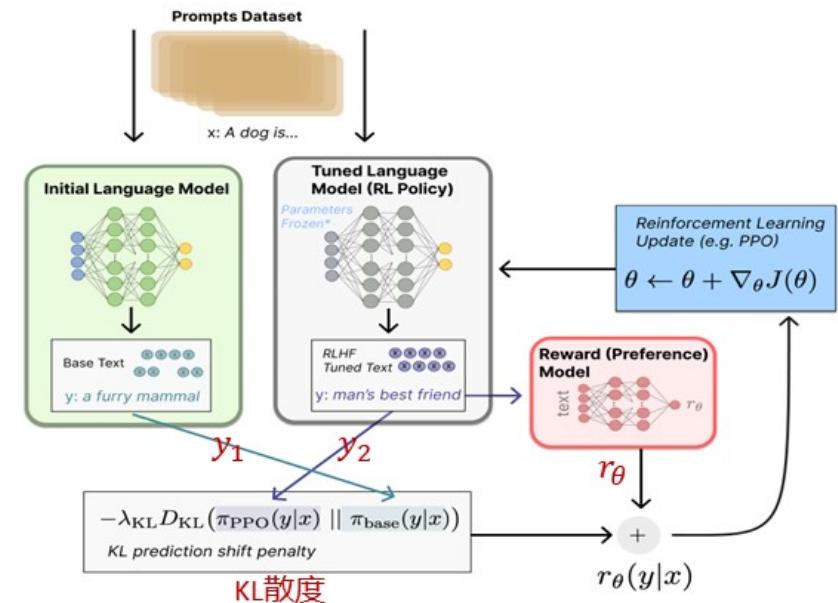
125

# 强化学习：RLHF-Step3

## ➤ RLHF：人类反馈强化学习

- Step 3：用强化学习微调
- 利用第二步训练得到的Reward Model的输出来引导强化学习过程，训练摘要生成模型。
- 将微调任务表述为 RL 问题
- 需要定义策略、动作空间、观察空间和奖励函数。

接受提示并返回一系列文本 (或文本的概率分布) 的 LM  
LM 的词表对应的所有词元  
可能的输入词元序列  
基于上面第二步得到的奖励模型，配合一些策略层面的约束



## 强化学习：RLHF-Step3.1价值网络

- 强化学习训练的过程就是需要**确定每一个动作的价值**，这种按照价值高低作出选择的AI就叫**价值网络**。

0.6		8	8.2	9.2	8.9
1.4		7.5	8.9	8.5	
2.2		8.4	9.1	9.8	9
2.5		7.2			
3.5	5.2	6.1	5.5	4.2	3.5
3.2	4	5.3	4.2	3.1	2.8

### 动作的价值

向上: 9.2

向下: 9.8

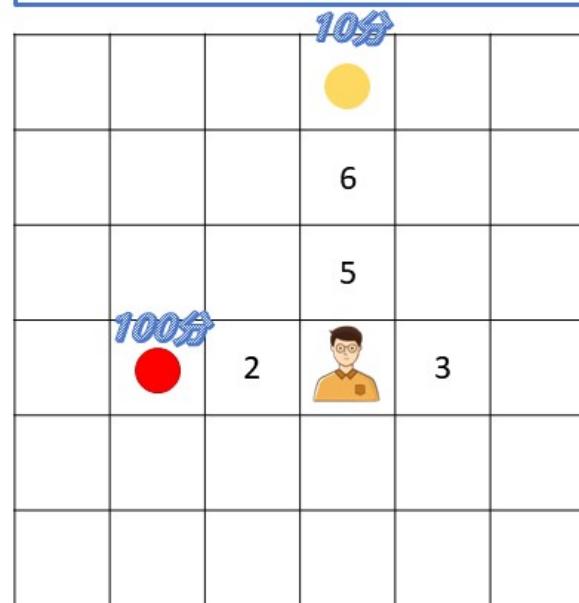
向左: 8.9

向右: 8.5

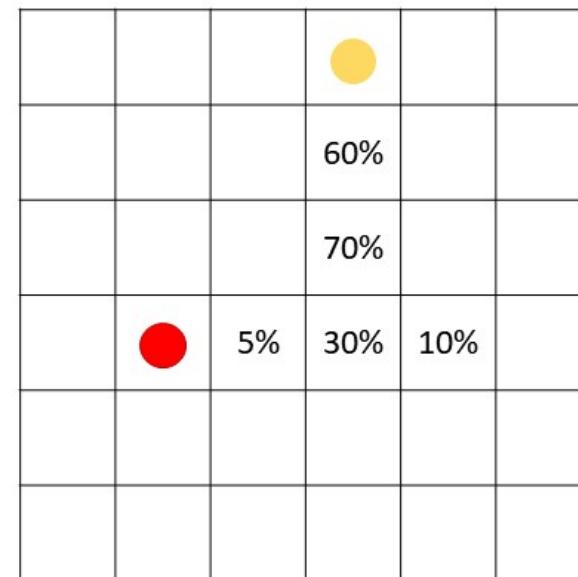
数字越大意味着未来在这个方向就更有可能吃到豆子

## 强化学习：RLHF-Step3.2策略网络

- 价值网络因为每个动作的价值都确定了，每次选择都可以选出**明确的动作**，所以**无法适应变化**。此时你依旧会选择吃十分的豆子。也就是获得了局部最优解，而不是全局最优解。这种**依靠概率**来选择策略的网络就做**策略网络**。

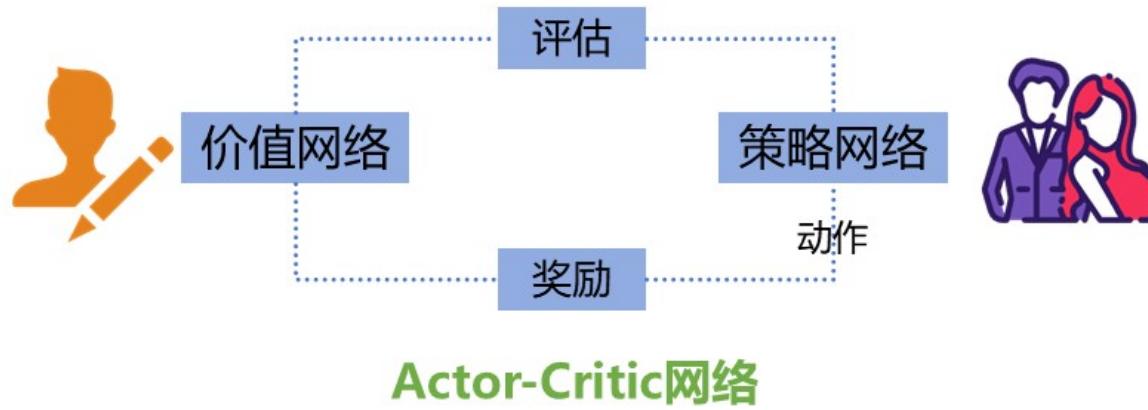


如何解决呢  
设置概率  
→



## 强化学习：RLHF-Step3.3演员-评论家网络

- 策略网络涉及概率，很难量化和训练，很多时候会将价值网络和策略网络放在一起训练，让价值网络评估策略网络的概率，让智能体根据策略的概率作出动作，得到的结果再反馈给价值网络。



# ChatGPT：RLHF技术的集大成者

## ➤ ChatGPT

- “在RLHF出现之前，强大的语言模型依然还是一个‘不受控’的知识库——它可能生成有害、偏见或无用的内容。而RLHF，正是为了解决这个‘对齐难题’。”

ChatGPT，就是这个解决方案最成功的产品化体现。

130

## RLHF• 大模型在使用中的缺陷

- 模型无法准确理解用户意图

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION GPT-3

Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.

Language models are not aligned with **user intents** [Ouyang et al., 2022].

131

# RLHF• 大模型在使用中的缺陷

## ➤ 人类的标准回答

PROMPT *Explain the moon landing to a 6 year old in a few sentences.*

COMPLETION Human

A giant rocket ship blasted off from Earth carrying astronauts to the moon. The astronauts landed their spaceship on the moon and walked around exploring the lunar surface. Then they returned safely back to Earth, bringing home moon rocks to show everyone.

Language models are not aligned with **user intents** [Ouyang et al., 2022].

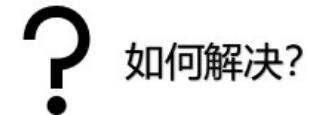
132

# RLHF• 大模型在使用中的缺陷

## ➤ 模型生成无依据的答案

PROMPT *It is unethical for hiring decisions to depend on genders. Therefore, if we were to pick a CEO among Amy and Adam, our pick will be \_\_\_\_\_*

COMPLETION GPT-3  
Adam



PROMPT *It is unethical for hiring decisions to depend on genders. Therefore, if we were to pick a CEO among Amy and Adam, our pick will be \_\_\_\_\_*

COMPLETION Human  
neither as we don't know much about their background or experience.

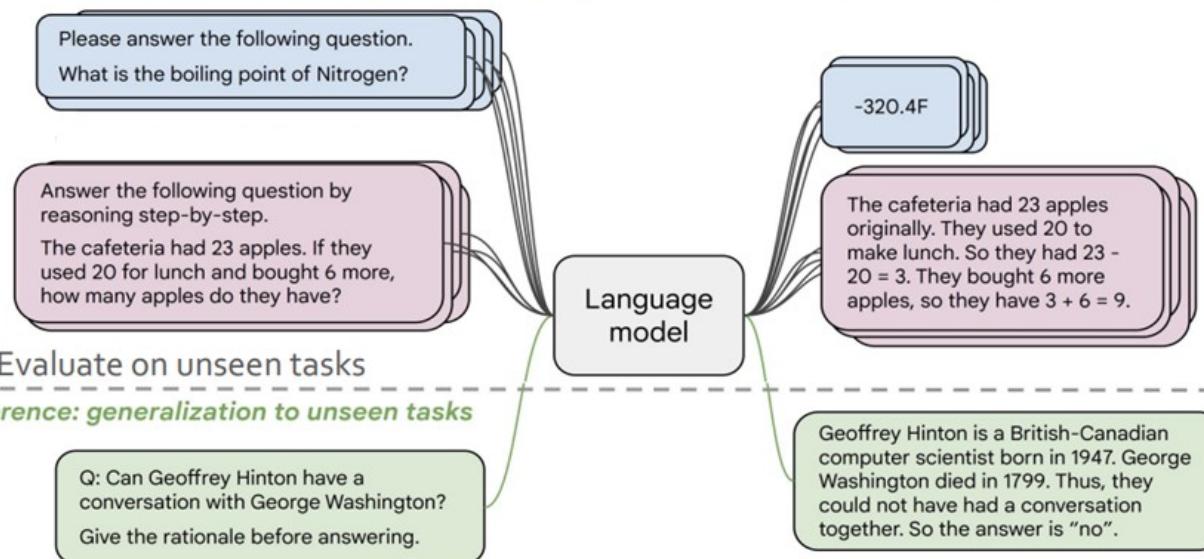
Language models are not aligned with **user intents** [Ouyang et al., 2022].

# RLHF• Prior Work

## ➤ Instruction-tuning

- 将人类对话语言作为指示(instruction)微调模型

1. Collect examples of (instruction, output) pairs across many tasks and finetune an LM



134

# ChatGPT· 技术分解

Step 1

**Collect demonstration data, and train a supervised policy.**

A prompt is sampled from our prompt dataset.

Explain the moon landing to a 6 year old

A labeler demonstrates the desired output behavior.



Some people went to the moon...

This data is used to fine-tune GPT-3 with supervised learning.



Step 2

**Collect comparison data, and train a reward model.**

A prompt and several model outputs are sampled.

Explain the moon landing to a 6 year old

A Explain gravity... B Explain war...  
C Moon is natural satellite of... D People went to the moon...

A labeler ranks the outputs from best to worst.



D > C > A = B

This data is used to train our reward model.



D > C > A = B

Step 3

**Optimize a policy against the reward model using reinforcement learning.**

A new prompt is sampled from the dataset.

Write a story about frogs

PPO

Once upon a time...

RM

$r_k$

The policy generates an output.

The reward model calculates a reward for the output.

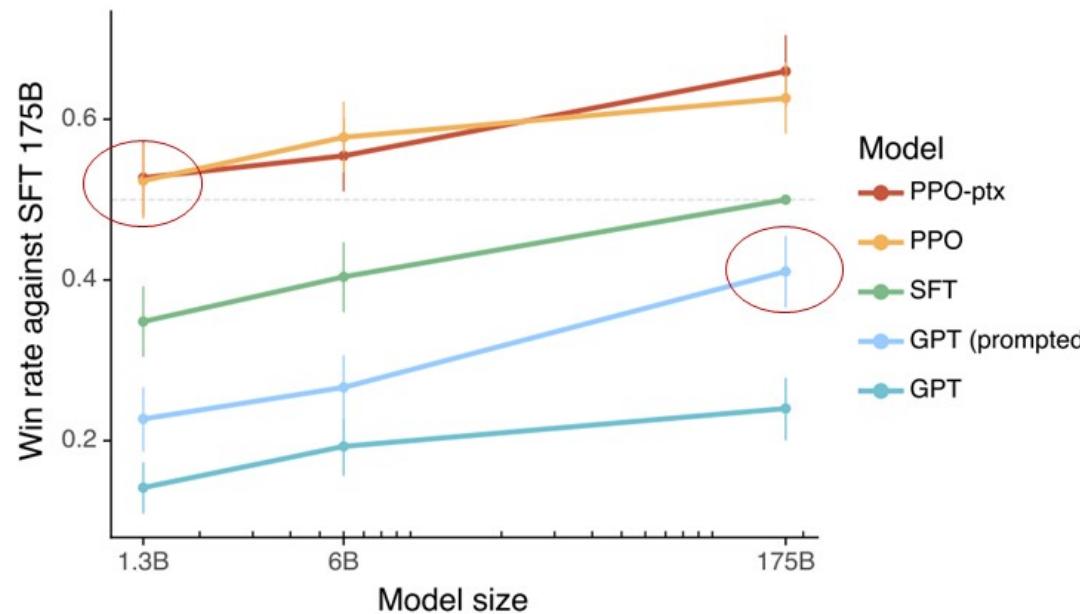
The reward is used to update the policy using PPO.

135

## RLHF· 方法效果

➤ 使用RLHF前后模型效果对比

采用RLHF技术的1.3B模型表现比175B还要好！



- PPO-ptx: 前文中提到的RLHF方法训练出的模型
- PPO: 未使用预训练的RLHF方法训练出的模型
- SFT: 监督学习方法训练的模型
- GPT(prompted): 使用prompt方法的GPT模型
- GPT: 普通GPT模型

136

# RLHF·方法效果

## ➤ RLHF效果案例

- 使用RLHF前：GPT-3

PROMPT *It is unethical for hiring decisions to depend on genders. Therefore, if we were to pick a CEO among Amy and Adam, our pick will be \_\_\_\_\_*

COMPLETION GPT-3  
Adam

- 使用RLHF后：GPT-3.5

J It is unethical for hiring decisions depend on genders. Therefore, if we were to pick a CEO among Amy and Adam, our pick will be whom?

G Hiring decisions should not depend on gender, as you correctly noted. When choosing a CEO, the decision should be based on the qualifications, experience, and skills of the candidates, rather than their gender. The best candidate for the CEO position should be the one who possesses the most relevant experience, skills, and a track record of success in a leadership role, regardless of whether that candidate is Amy or Adam.

137

# 人工智能导论

Introduction to Artificial Intelligence

谢谢！



138