

# 大语言模型构建、评测和提示学习

兰韵诗

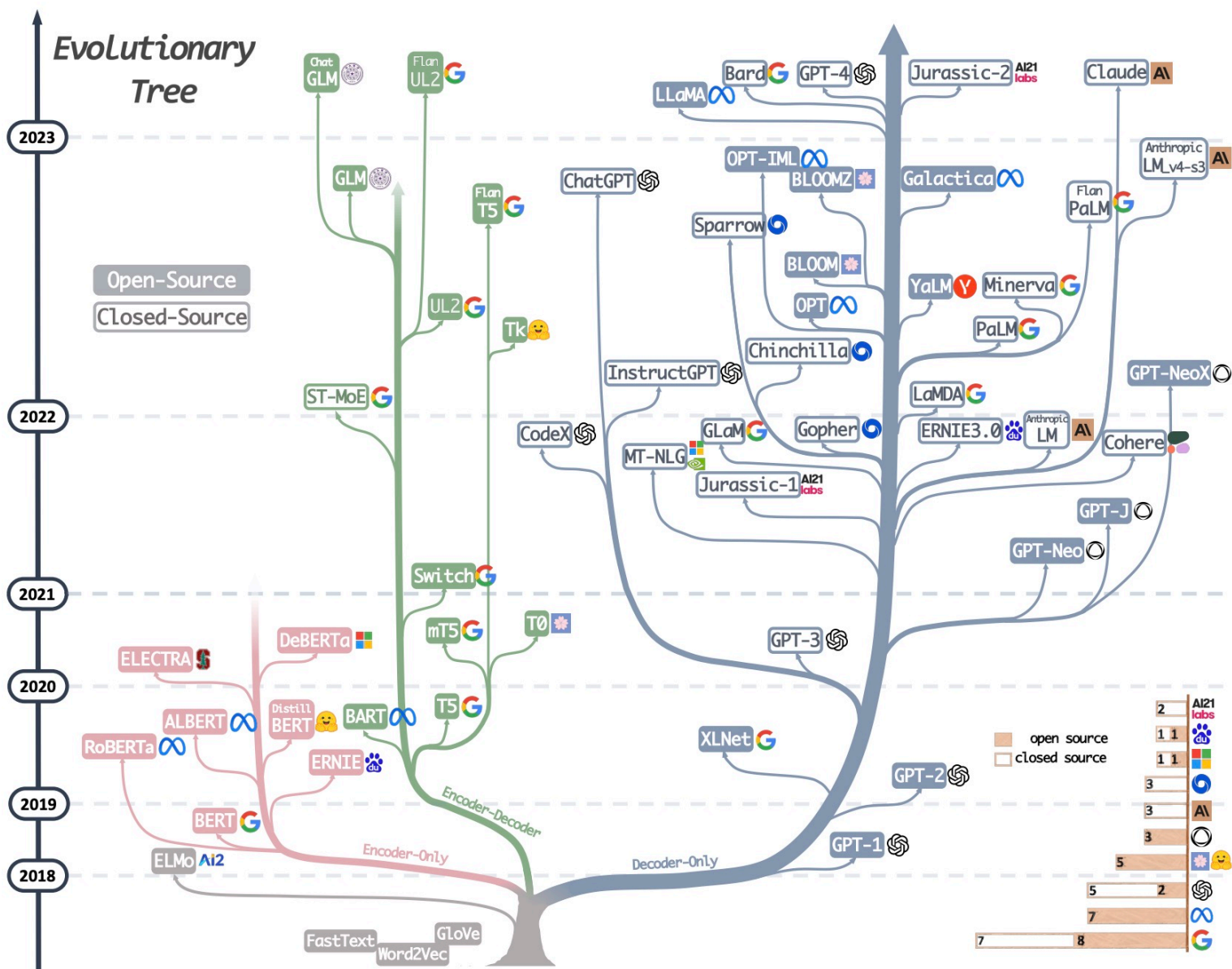
2025-05

- 大语言模型的构建
- 提示学习
- 大语言模型的评测

- 《大语言模型》第2、10、12章

- 大语言模型的构建
- 提示学习
- 大语言模型的评测

# 预训练语言模型发展史



- 大语言模型是指在海量无标注文本数据上，进行预训练得到的大型预训练语言模型，如GPT-3, PaLM和LLaMA。
- 目前大语言模型所需要具有的最小参数规模还没有一个明确的参考标准，但是大语言模型通常是指参数规模达到百亿、千亿甚至万亿的模型。
- 也有部分工作认为经过大规模数据预训练（显著多于传统预训练模型如BERT所需要的训练数据）的数十亿参数级别的模型也可以大语言模型（如LLaMA-2.7B）。

- 和传统语言模型相比，大语言模型的构建过程设计到更为复杂的训练方法，进而展现出了强大的自然语言理解能力和复杂任务求解能力。

- 大规模预训练
  - 使用与下游任务无关的大规模数据进行模型参数的初始训练。
- 指令微调与人类对齐
  - 经过大规模数据预训练后的语言模型已经具备较强的模型能力。
  - 但由于预训练任务形式所限，这些模型更擅长于文本补全，并不适合解决具体的任务。
  - 指令微调(Supervised Fine-Tuning, SFT), 通过使用任务输入与输出的配对数据进行模型训练，使语言模型较好地掌握通过问答形式进行任务求解的能力。
  - 人类对齐(Reinforcement Learning from Human Feedback, RLHF), 引入了基于人类反馈的强化学习对齐方法。

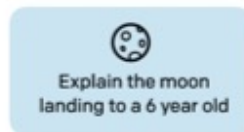


# 大语言模型的构建

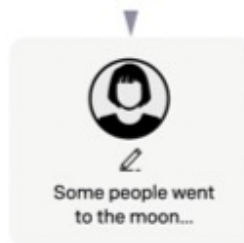
## Step 1

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

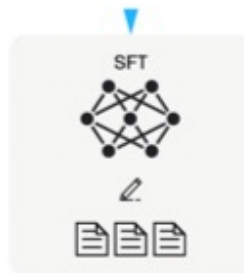
A prompt is  
sampled from our  
prompt dataset.



A labeler  
demonstrates the  
desired output  
behavior.



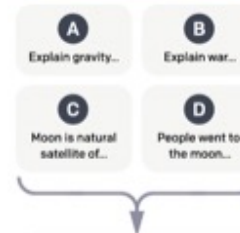
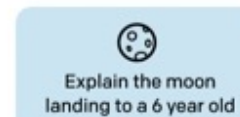
This data is used  
to fine-tune GPT-3  
with supervised  
learning.



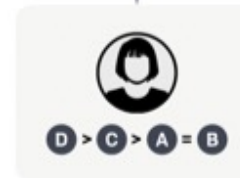
## Step 2

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

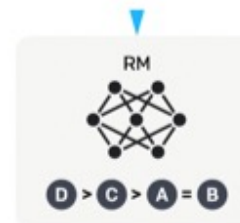
A prompt and  
several model  
outputs are  
sampled.



A labeler ranks  
the outputs from  
best to worst.



This data is used  
to train our  
reward model.



## Step 3

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

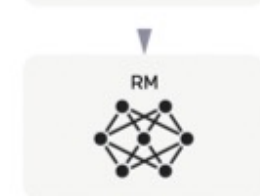
A new prompt  
is sampled from  
the dataset.



The policy  
generates  
an output.



The reward model  
calculates a  
reward for  
the output.

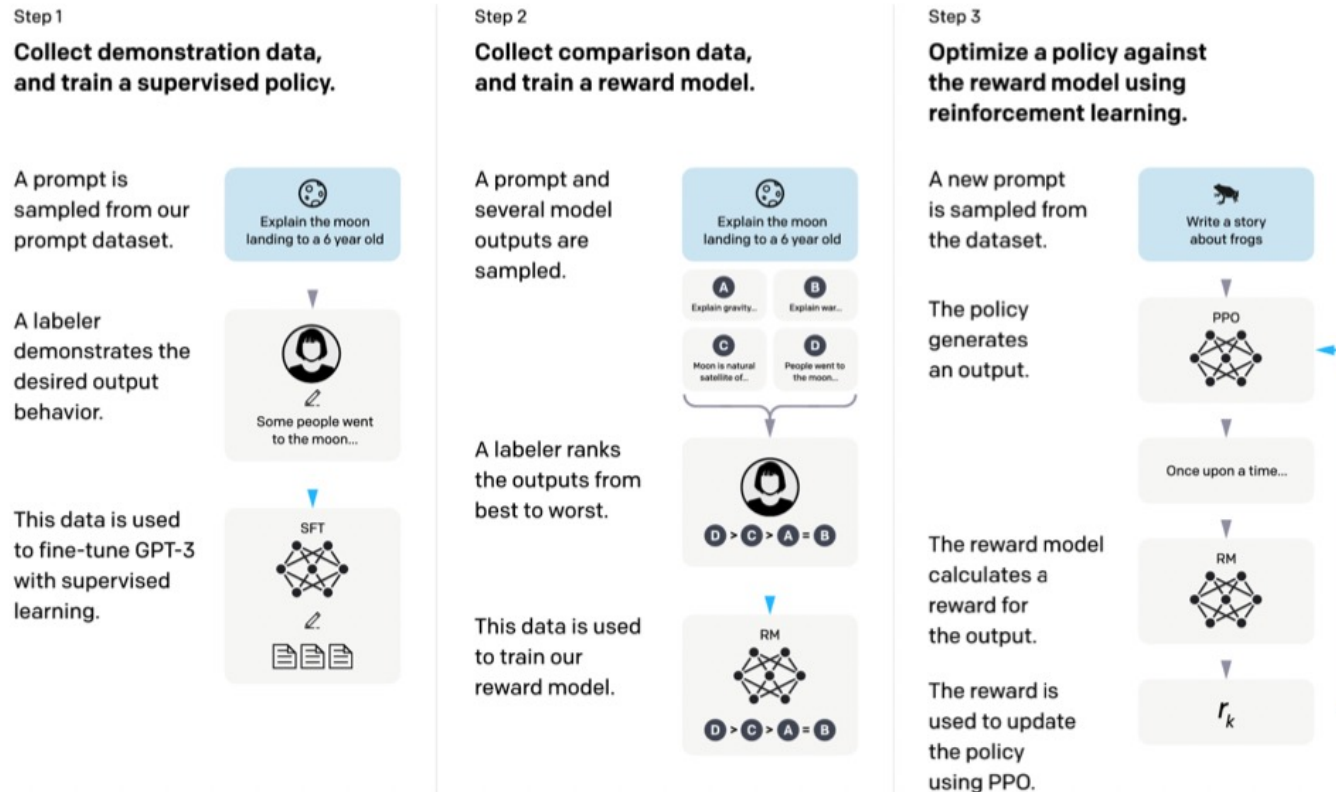


The reward is  
used to update  
the policy  
using PPO.



Training language models to follow instructions with human feedback. Ouyang et al. 2022.

# 大语言模型的构建



指令微调

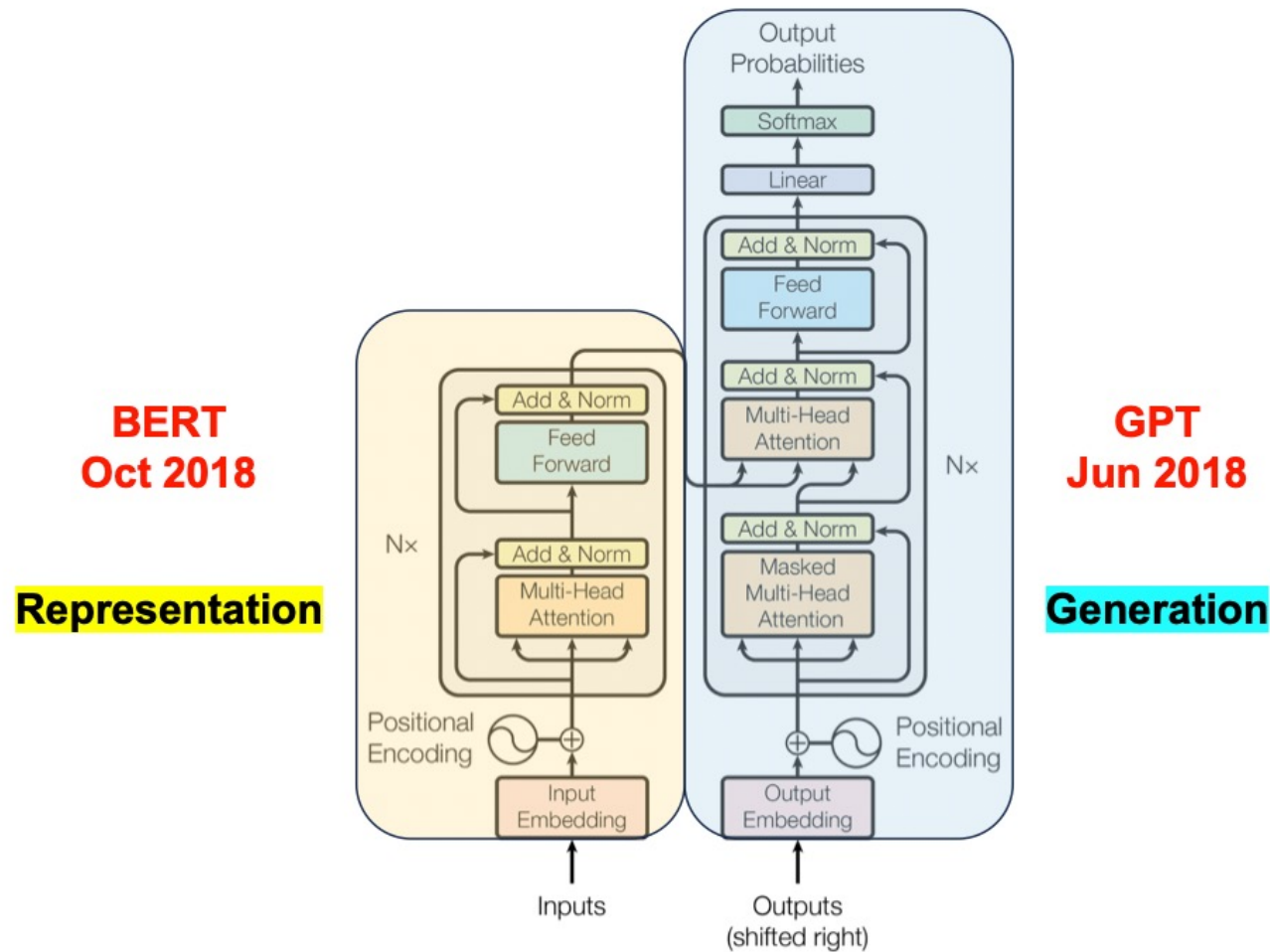
人类对齐

Training language models to follow instructions with human feedback. Ouyang et al. 2022.

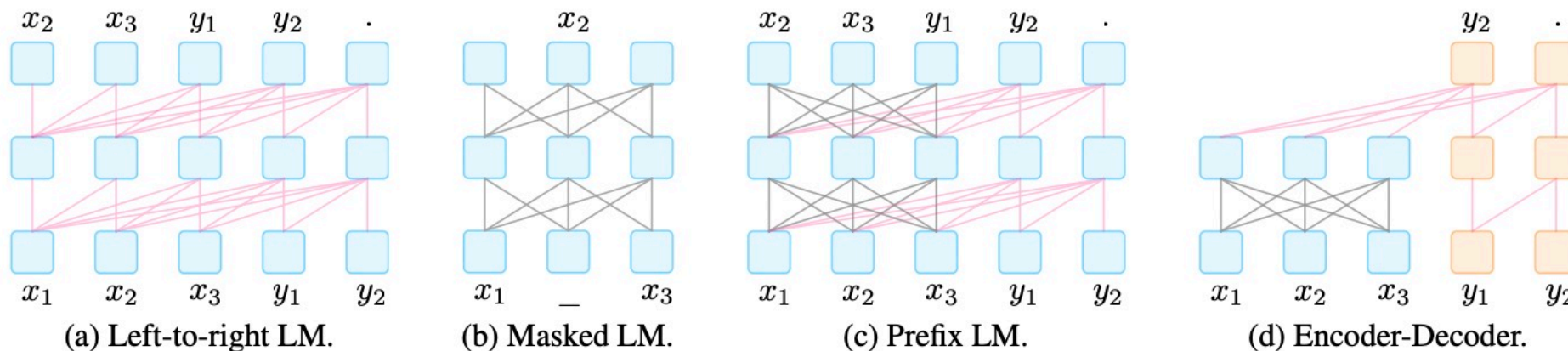
可公开获取 模型	发布 时间	大小 (B)	适配 IT RLHF	预训练 数据规模	硬件 (GPUs / TPUs)	训练 时间	
T5	2019.10	11	-	-	1000B 词元	1024 TPU v3	-
CodeGen	2022.03	16	-	-	577B 词元	-	-
OPT	2022.05	175	-	-	180B 词元	992 A100 (80G)	-
CodeGeeX	2022.09	13	-	-	850B 词元	1536 Ascend 910	60 天
GLM	2022.10	130	-	-	400B 词元	768 A100 (40G)	60 天
BLOOM	2022.11	176	✓	-	366B 词元	384 A100 (80G)	105 天
Galactica	2022.11	120	-	-	106B 词元	-	-
LLaMA	2023.02	65	-	-	1400B 词元	2048 A100 (80G)	21 天
Pythia	2023.04	12	-	-	300B 词元	256 A100 (40G)	-
CodeGen-2	2023.05	16	-	-	400B 词元	-	-
StarCoder	2023.05	15.5	-	-	1000B 词元	512 A100 (40G)	-
Falcon	2023.06	180	-	-	3500B 词元	4096 A100 (40G)	-
LLaMA-2	2023.07	70	✓	✓	2000B 词元	2000 A100 (80G)	-
Baichuan-2	2023.09	13	✓	✓	2600B 词元	1024 A800	-
QWEN	2023.09	14	✓	✓	3000B 词元	-	-
FLM	2023.09	101	✓	-	311B 词元	192 A800	22 天
Mistral	2023.09	7	✓	-	-	-	-
Skywork	2023.10	13	-	-	3200B 词元	512 A800 (80G)	-
Mixtral	2023.12	47	✓	-	-	-	-
DeepSeek	2024.01	67	✓	✓	2000B 词元	-	-

闭源模型	发布时间	大小 (B)	适配		预训练 数据规模	硬件 (GPUs / TPUs)	训练 时间
			IT	RLHF			
GPT-3	2020.05	175	-	-	300B 词元	-	-
Codex	2021.07	12	-	-	100B 词元	-	-
ERNIE 3.0	2021.07	10	-	-	375B 词元	384 V100	-
FLAN	2021.09	137	✓	-	-	128 TPU v3	60 小时
Yuan 1.0	2021.10	245	-	-	180B 词元	2128 GPU	-
Anthropic	2021.12	52	-	-	400B 词元	-	-
WebGPT	2021.12	175	-	✓	-	-	-
Gopher	2021.12	280	-	-	300B 词元	4096 TPU v3	920 小时
LaMDA	2022.01	137	-	-	768B 词元	1024 TPU v3	57.7 天
MT-NLG	2022.01	530	-	-	270B 词元	4480 A100 (80G)	-
AlphaCode	2022.02	41	-	-	967B 词元	-	-
InstructGPT	2022.03	175	✓	✓	-	-	-
Chinchilla	2022.03	70	-	-	1400B 词元	-	-
PaLM	2022.04	540	-	-	780B 词元	6144 TPU v4	-
GPT-4	2023.03	-	✓	✓	-	-	-
PanGu-Σ	2023.03	1085	-	-	329B 词元	512 Ascend 910	100 天
PaLM-2	2023.05	16	✓	-	100B 词元	-	-

## Review: The LLM Era – Paradigm Shift in Machine Learning







## 单向（因果）语言模型

代表：  
ELMO、SiATL  
GPT、GPT-2、  
GPT-3、GPT-4、  
GPT-J、OPT、  
ChatGPT、LLaMA  
、Alpaca

## 双向（掩码）语言模型

代表：  
BERT、MT-DNN、ELECTRA、  
SpanBERT、RoBERTa、ERNIE-  
THU、ERNIE-Baidu-1.0/2.0/3.0  
ALBERT、XLNet、DeBERTa、  
SpellBERT、MacBERT、CoLAKE  
、StructBERT、K-BERT、  
KEPLER、DKPLM、LayoutLM-  
V1/2/3、Longformer、Roformer

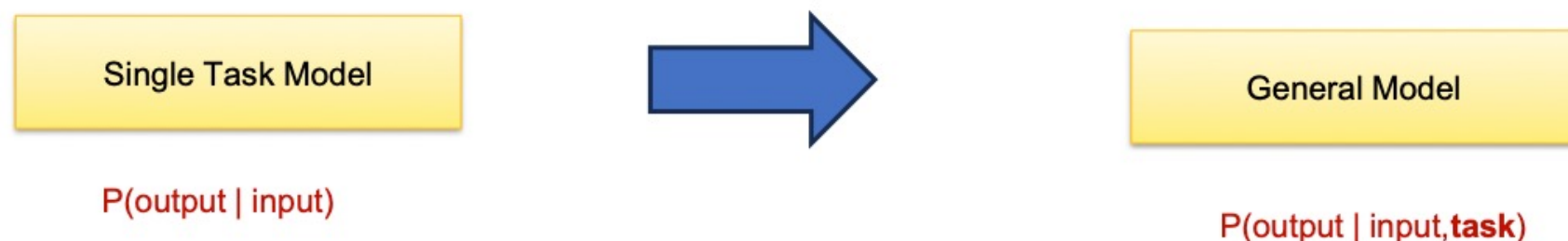
## 混合自回归

代表：  
UniLM、GLM、  
M6、BiET、VLMo

## 端到端语言模型

代表：  
BART、T5、MASS

- GPT2: 微调和人类对齐可以是多任务的



- 任务定义均需要提供成文本的形式

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	<b>21.8</b>
117M	<b>35.13</b>	45.99	<b>87.65</b>	<b>83.4</b>	<b>29.41</b>	65.85	1.16	1.17	37.50	75.20
345M	<b>15.60</b>	55.48	<b>92.35</b>	<b>87.1</b>	<b>22.76</b>	47.33	1.01	<b>1.06</b>	26.37	55.72
762M	<b>10.87</b>	<b>60.12</b>	<b>93.45</b>	<b>88.0</b>	<b>19.93</b>	<b>40.31</b>	<b>0.97</b>	<b>1.02</b>	22.05	44.575
1542M	<b>8.63</b>	<b>63.24</b>	<b>93.30</b>	<b>89.05</b>	<b>18.34</b>	<b>35.76</b>	<b>0.93</b>	<b>0.98</b>	<b>17.48</b>	42.16

- 困惑度 ( Perplexity ) 是信息论的一个概念，可以用来衡量一个分布的不确定性。
- 给定一个测试文本集合，一个好的序列生成模型应该使得测试集中的句子的联合概率尽可能高。
- 困惑度可以衡量模型分布与样本经验分布之间的契合程度。困惑度越低则两个分布越接近。

假设测试集合共有独立同分布的  $N$  个序列  $\{\mathbf{x}_{1:T_n}^{(n)}\}_{n=1}^N$ 。我们可以用模型  $p_\theta(\mathbf{x})$  对每个序列计算其概率  $p_\theta(\mathbf{x}_{1:T_n}^{(n)})$ ，整个测试集的联合概率为

$$\prod_{n=1}^N p_\theta(\mathbf{x}_{1:T_n}^{(n)}) = \prod_{n=1}^N \prod_{t=1}^{T_n} p_\theta(x_t^{(n)} | \mathbf{x}_{1:(t-1)}^{(n)}). \quad (15.35)$$

模型  $p_\theta(\mathbf{x})$  的困惑度定义为

$$\text{PPL}(\theta) = 2^{-\frac{1}{T} \sum_{n=1}^N \log p_\theta(\mathbf{x}_{1:T_n}^{(n)})} \quad (15.36)$$

$$= 2^{-\frac{1}{T} \sum_{n=1}^N \sum_{t=1}^{T_n} \log p_\theta(x_t^{(n)} | \mathbf{x}_{1:(t-1)}^{(n)})} \quad (15.37)$$

$$= \left( \prod_{n=1}^N \prod_{t=1}^{T_n} p_\theta(x_t^{(n)} | \mathbf{x}_{1:(t-1)}^{(n)}) \right)^{-1/T}, \quad (15.38)$$

其中  $T = \sum_{n=1}^N T_n$  为测试数据集中序列的总长度。可以看出，困惑度为每个词条件概率  $p_\theta(x_t^{(n)} | \mathbf{x}_{1:(t-1)}^{(n)})$  的几何平均数的倒数。测试集中所有序列的概率越大，困惑度越小，模型越好。



- BLEU ( Bilingual Evaluation Understudy ) 是衡量模型生成序列和参考序列之间的N元词组 ( N-Gram ) 的重合度，最早用来评价机器翻译模型的质量，目前也广泛应用在各种序列生成任务中。

假设从模型分布  $p_\theta$  中生成一个候选 (Candidate) 序列  $\mathbf{x}$ , 从真实数据分布中采样出的一组参考 (Reference) 序列  $\mathbf{s}^{(1)}, \dots, \mathbf{s}^{(K)}$ , 我们首先从生成序列中提取 N 元组合的集合  $\mathcal{W}$ , 并计算 N 元组合的精度 (Precision),

$$P_n(\mathbf{x}) = \frac{\sum_{w \in \mathcal{W}} \min(c_w(\mathbf{x}), \max_{k=1}^K c_w(\mathbf{s}^{(k)}))}{\sum_{w \in \mathcal{W}} c_w(\mathbf{x})}, \quad (15.39)$$

其中  $c_w(\mathbf{x})$  是 N 元组合  $w$  在生成序列  $\mathbf{x}$  中出现的次数,  $c_w(\mathbf{s}^{(k)})$  是 N 元组合  $w$  在参考序列  $\mathbf{s}^{(k)}$  中出现的次数。N 元组合的精度  $P_n(\mathbf{x})$  是计算生成序列中的 N 元组合有多少比例在参考序列中出现。

BLEU 是通过计算不同长度的 N 元组合的精度, 并进行几何加权平均而得到。

$$\text{BLEU-N}(\mathbf{x}) = b(\mathbf{x}) \times \exp\left(\sum_{n=1}^N w_n \log P_n\right), \quad (15.41)$$

$$b(\mathbf{x}) = \begin{cases} 1 & \text{if } l_x > l_s \\ \exp(1 - l_s/l_x) & \text{if } l_x \leq l_s \end{cases}$$

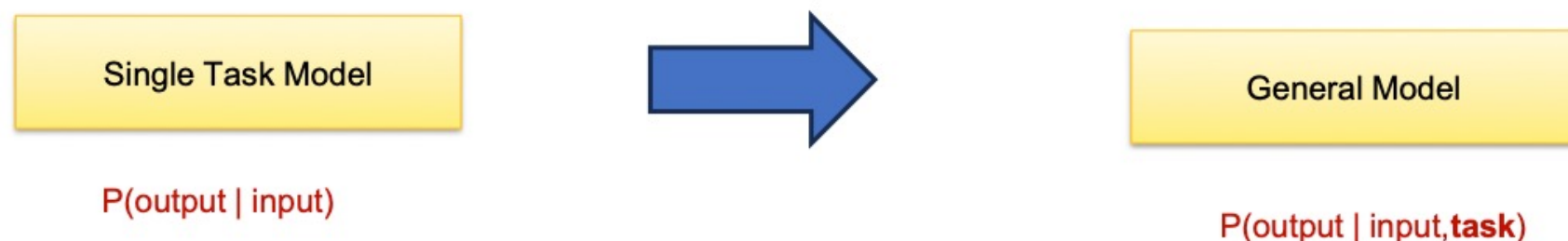
- ROUGE ( Recall-Oriented Understudy for Gisting Evaluation ) 最早应用于文本摘要领域。和 BLEU类似，但ROUGE计算的是召回率 ( Recall ) 。

假设从模型分布  $p_\theta$  中生成一个候选 (Candidate) 序列  $\mathbf{x}$ ，从真实数据分布中采样出的一组参考 (Reference) 序列  $\mathbf{s}^{(1)}, \dots, \mathbf{s}^{(K)}$ ，令  $\mathcal{W}$  为从参考序列中提取 N 元组合的集合，ROUGE-N 的定义为

$$\text{ROUGE-N}(\mathbf{x}) = \frac{\sum_{k=1}^K \sum_{w \in \mathcal{W}} \min \left( c_w(\mathbf{x}), c_w(\mathbf{s}^{(k)}) \right)}{\sum_{k=1}^K \sum_{w \in \mathcal{W}} c_w(\mathbf{s}^{(k)})}, \quad (15.42)$$

其中  $c_w(\mathbf{x})$  是 N 元组合  $w$  在生成序列  $\mathbf{x}$  中出现的次数， $c_w(\mathbf{s}^{(k)})$  是 N 元组合  $w$  在参考序列  $\mathbf{s}^{(k)}$  中出现的次数。

- GPT2: 微调和人类对齐可以是多任务的



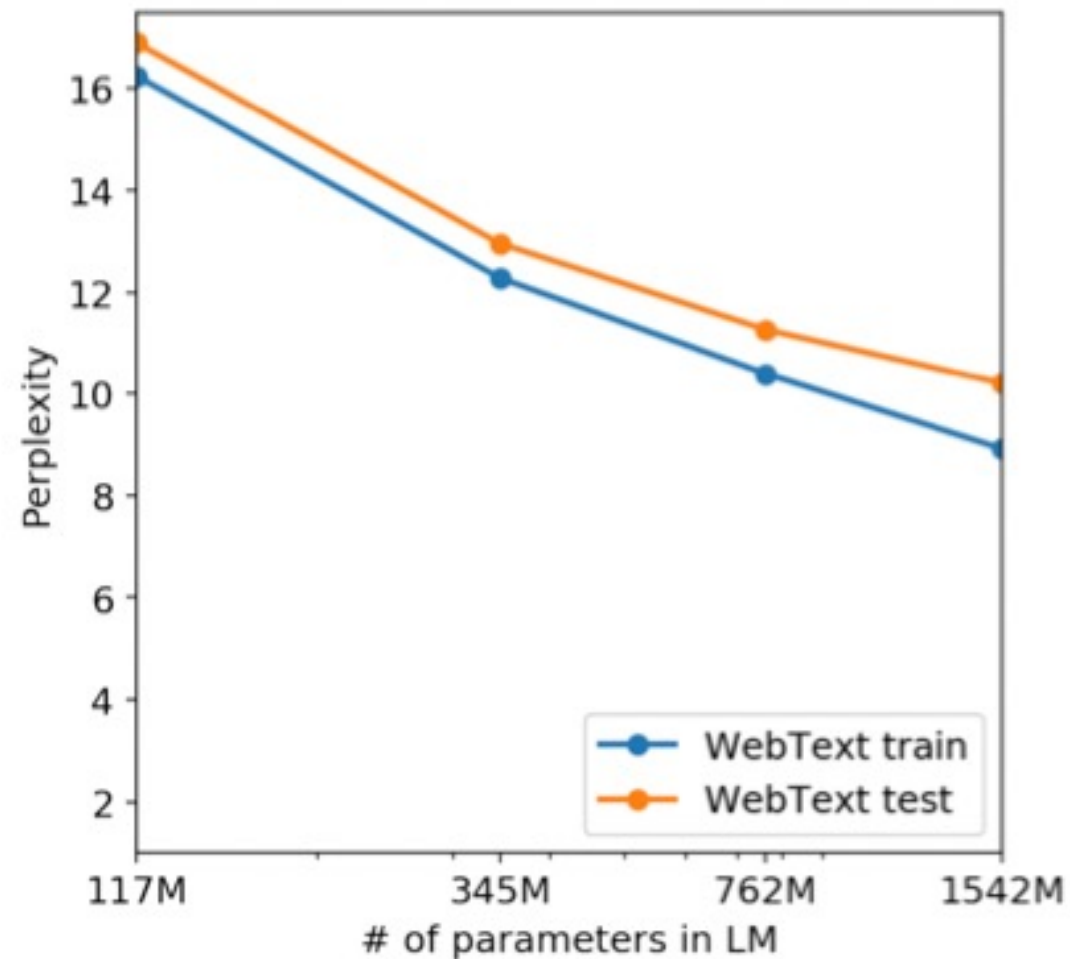
- 任务定义均需要提供成文本的形式

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	<b>21.8</b>
117M	<b>35.13</b>	45.99	<b>87.65</b>	<b>83.4</b>	<b>29.41</b>	65.85	1.16	1.17	37.50	75.20
345M	<b>15.60</b>	55.48	<b>92.35</b>	<b>87.1</b>	<b>22.76</b>	47.33	1.01	<b>1.06</b>	26.37	55.72
762M	<b>10.87</b>	<b>60.12</b>	<b>93.45</b>	<b>88.0</b>	<b>19.93</b>	<b>40.31</b>	<b>0.97</b>	<b>1.02</b>	22.05	44.575
1542M	<b>8.63</b>	<b>63.24</b>	<b>93.30</b>	<b>89.05</b>	<b>18.34</b>	<b>35.76</b>	<b>0.93</b>	<b>0.98</b>	<b>17.48</b>	42.16

- 大语言模型采用了与小型预训练语言模型相似的神经网络结构（基于注意力机制的transformer架构）和预训练方法（如语言建模）。但是用过扩展参数规模、数据规模和计算算力，大语言模型的能力显著超过了小型语言模型的能力。而这些能力显著高于通过改进架构、算法等方面所带来的改进。

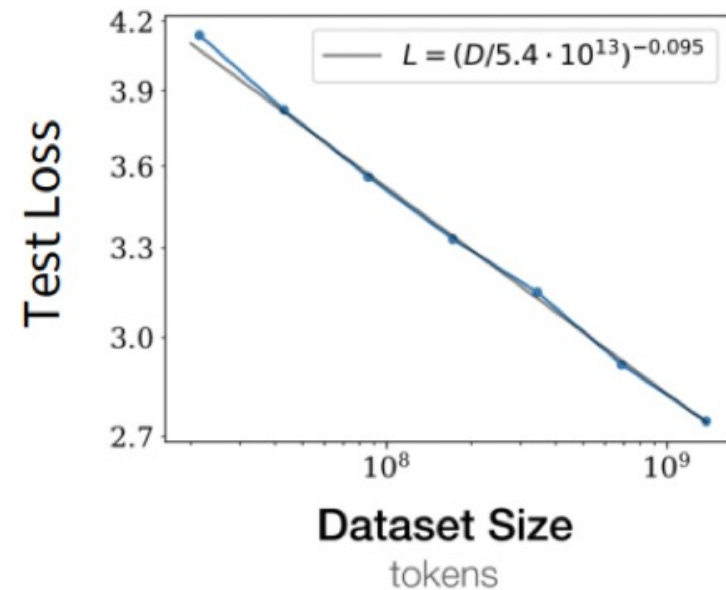
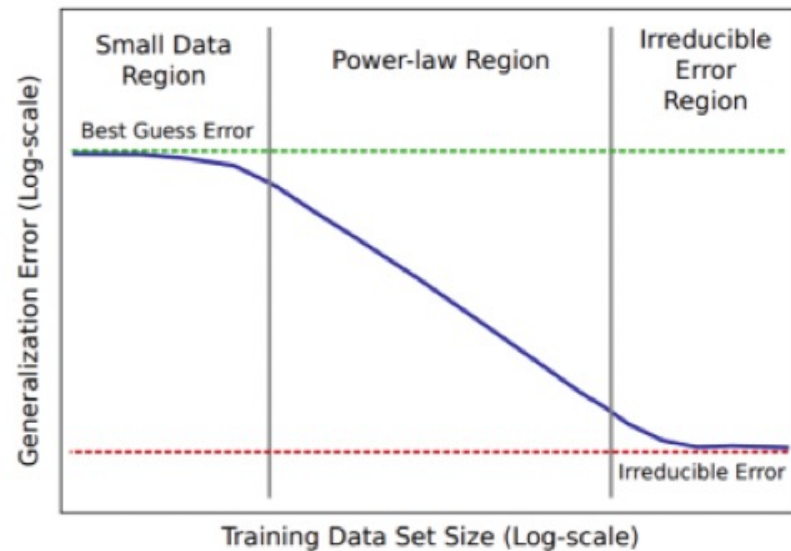
# 大模型的扩展法则

- GPT2: 扩展提高语言模型的困惑度和效果



# 大模型的扩展法则

- “power law scaling”





- 我们能够将扩展法则通过数据表达式体现出来？

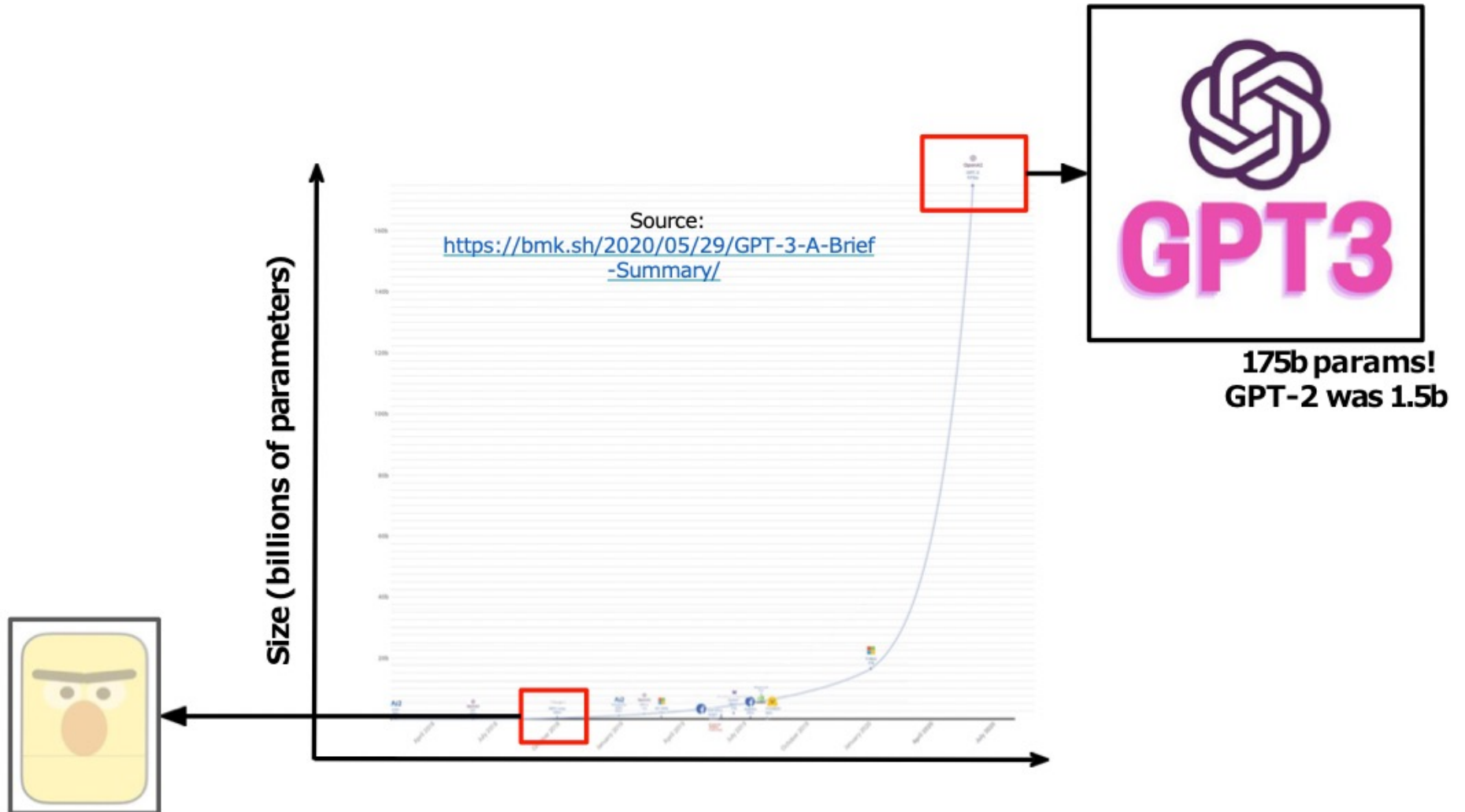
2020 年，Kaplan 等人 [15] (OpenAI 团队) 首次建立了神经语言模型性能与三个主要因素——模型规模 ( $N$ )、数据规模 ( $D$ ) 和计算算力 ( $C$ ) 之间的幂律关系 (Power-Law Relationship)。由于原始论文中没有给出具体的扩展法则命名，本部分内容中使用两位共同第一作者姓氏的首字母来进行命名。在给定算力预算  $c$  的条件下，可以近似得到以下三个基本指数公式来描述扩展法则：

$$\begin{aligned} L(N) &= \left( \frac{N_c}{N} \right)^{\alpha_N}, \quad \alpha_N \sim 0.076, N_c \sim 8.8 \times 10^{13} \\ L(D) &= \left( \frac{D_c}{D} \right)^{\alpha_D}, \quad \alpha_D \sim 0.095, D_c \sim 5.4 \times 10^{13} \\ L(C) &= \left( \frac{C_c}{C} \right)^{\alpha_C}, \quad \alpha_C \sim 0.050, C_c \sim 3.1 \times 10^8 \end{aligned} \quad (2.1)$$

这里， $L(\cdot)$  表示用以  $\text{nat}^1$  为单位的交叉熵损失。其中， $N_c$ 、 $D_c$  和  $C_c$  是实验性的常数数值，分别对应于非嵌入参数数量、训练数据数量和实际的算力开销。为了



## Model Scaling: GPT-3



## GPT-3 Pre-training – Architecture

Model Name	$n_{\text{params}}$	$n_{\text{layers}}$	$d_{\text{model}}$	$n_{\text{heads}}$	$d_{\text{head}}$	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	$6.0 \times 10^{-4}$
GPT-3 Medium	350M	24	1024	16	64	0.5M	$3.0 \times 10^{-4}$
GPT-3 Large	760M	24	1536	16	96	0.5M	$2.5 \times 10^{-4}$
GPT-3 XL	1.3B	24	2048	24	128	1M	$2.0 \times 10^{-4}$
GPT-3 2.7B	2.7B	32	2560	32	80	1M	$1.6 \times 10^{-4}$
GPT-3 6.7B	6.7B	32	4096	32	128	2M	$1.2 \times 10^{-4}$
GPT-3 13B	13.0B	40	5140	40	128	2M	$1.0 \times 10^{-4}$
GPT-3 175B or “GPT-3”	175.0B	96	12288	96	128	3.2M	$0.6 \times 10^{-4}$

**Table 2.1:** Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

## GPT-3 Pre-training – Training Data

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

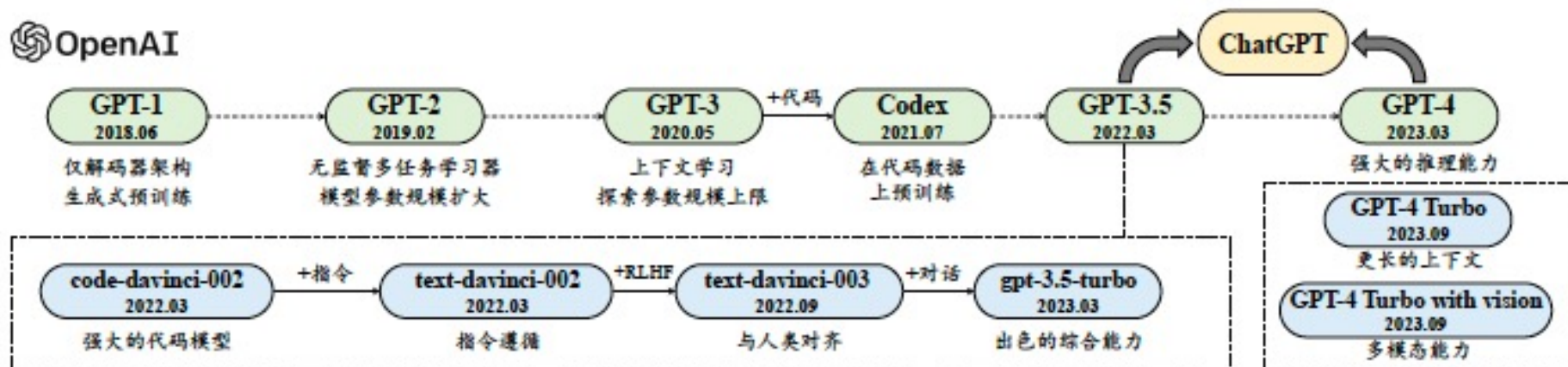
**Table 2.2: Datasets used to train GPT-3.** “Weight in training mix” refers to the fraction of examples during training that are drawn from a given dataset, which we intentionally do not make proportional to the size of the dataset. As a result, when we train for 300 billion tokens, some datasets are seen up to 3.4 times during training while other datasets are seen less than once.

- 大语言模型的涌现能力被非形式化定义为“在小型模型中不存在但在大模型中出现的能力”。
- 具体指当模型扩展到一定规模时，模型的特定任务性能突然出现显著跃升的趋势，远超随机水平。

- Prompt-tuning ( 提示学习 )
  - In-Context Learning ( 上下文学习 )
  - Chain-of-Thought ( 思维链 )

# 大模型的扩展法则和涌现能力

- 扩展法则和涌现能力提供了两种不同观点来理解大模型相对于小模型的优势，但是刻画了较为不同的扩展效应趋势。
- 扩展法则使用语言建模损失来衡量语言模型的整体性能，整体上展现出了较为平滑的性能提升趋势，具有较好可预测性。
- 涌现能力使用任务性能来衡量模型性能，整体上展现出随机规模扩展的骤然跃升趋势。



- 大语言模型的构建
- 提示学习
- 大语言模型的评测

- 设计合适的提示（prompting），通过自然语言接口与大模型进行交互。
- 提示的设计主要依靠人工设计和自动优化两种策略实现。
- 为了更好的解决未见过的任务，一种典型的提示方法是上下文学习（in-context learning），它将任务描述与示例以自然语言文本形式加入到提示中
- 思维链提示（chain-of-thought）作为一种增强技术，将一系列中间推理步骤加入到提示中，以增强复杂推理任务的解决效果。



**基于任务示例回答问题：**

**示例问题：**罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

**示例回答：**罗杰一开始有 5 个球，2 罐每罐 3 个网球就是 6 个网球。 $5 + 6 = 11$ 。  
答案是 11。

**用户输入：**袋子里有 16 个球，其中一半是高尔夫球，并且所有高尔夫球中有一半是蓝色的球。请问有多少个蓝色高尔夫球？

**模型输出：**袋子里有 16 个球，其中一半是高尔夫球。因此有  $16 / 2 = 8$  个高尔夫球。一半的高尔夫球是蓝色的。因此有  $8 / 2 = 4$  个蓝色高尔夫球。答案是 4。

**例 10.3** 提示设计中上下文信息的例子（灰色文字表示模型生成结果）

## 解答数学问题：

**用户输入：**袋子里有 16 个球，其中一半是高尔夫球，并且所有高尔夫球中有一半是蓝色的球。请问有多少个蓝色高尔夫球？

**模型输出：** 让我们一步一步地思考。

袋子里一共有 16 个球，其中一半是高尔夫球，也就是说，一共有 8 个高尔夫球。所有高尔夫球中有一半是蓝色的，也就是说有 4 个蓝色的高尔夫球。

## 推荐旅游景点：

**用户输入：**你是一名旅行向导。我会把我的位置告诉你，你需要向我推荐附近的旅游景点。我的第一个请求是“我在伊斯坦布尔，我只想参观博物馆”。

**模型输出：** 当然，以下是一些您可以考虑参观的博物馆：

1. 伊斯坦布尔考古博物馆：这个博物馆位于苏丹艾哈迈德广场，是土耳其最重要的考古博物馆之一。它展示了来自不同历史时期的珍贵文物，包括古代近东、古希腊和古罗马的艺术品。
2. 伊斯坦布尔现代艺术博物馆：位于博斯普鲁斯海峡畔的卡拉库埃斯梅斯地区，这个博物馆致力于展示当代土耳其和国际艺术。它拥有丰富的永久收藏和临时展览，涵盖绘画、雕塑、摄影、视频艺术和装置艺术等多个领域。
3. 圣索菲亚博物馆：这座博物馆位于伊斯坦布尔历史中心的圣索菲亚大教堂内，展示了拜占庭和奥斯曼帝国时期的艺术和文物。您可以在这里欣赏到壮丽的拜占庭建筑和壁画，以及众多的历史文物。

### 例 10.4 提示设计中提示策略的例子

# 提示设计原则

---

- 清晰地表达任务目标
- 分解为简单且详细的子任务
- 提供少样本示例
- 采用模型友好的提示格式

形式上，我们使用  $D_k = \{f(x_1, y_1), \dots, f(x_k, y_k)\}$  来表示由  $k$  个样本构成的一组示例数据，其中  $f(x_k, y_k)$  是一个函数，负责将第  $k$  个任务样本转换为自然语言提示。给定任务描述  $I$ 、示例  $D_k$  以及新的输入  $x_{k+1}$ ，大语言模型生成答案  $\hat{y}_{k+1}$  的过程可以通过下面的公式来表述：

$$\underbrace{\text{LLM}}_{\text{大语言模型}} \left( \underbrace{I}_{\text{任务描述}}, \underbrace{f(x_1, y_1), \dots, f(x_k, y_k)}_{\text{示例}}, \underbrace{f(x_{k+1}, \quad)}_{\text{输入}}, \underbrace{\quad}_{\text{答案}} \right) \rightarrow \hat{y}_{k+1}. \quad (10.1)$$



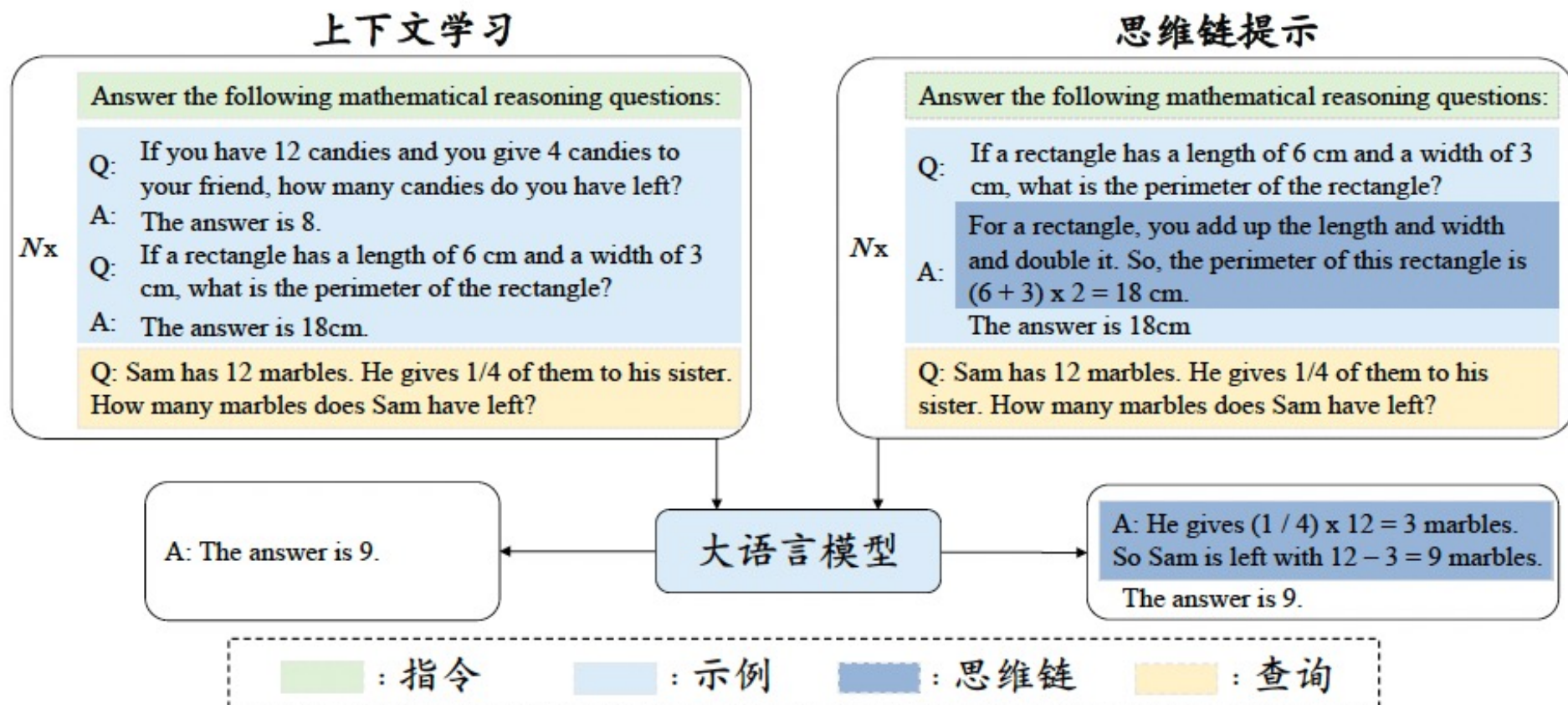


图 10.1 一个关于上下文学习和思维链提示的比较说明 (图片来源: [10])

# 上下文学习优化策略

---

- 基于相关度排序的方法
- 基于集合多样性的方法
- 基于大语言模型的方法

# 上下文学习

## (1) 包含输入与输出的示例格式：

输入：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

输出：11。

示例模板：问题：{输入} 答案：{输出}

具体示例：问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？答案：11。

## (2) 增加任务描述的示例格式：

输入：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

输出：11。

示例模板：下面是一个小学数学问题。问题：{输入} 答案：{输出}

具体示例：下面是一个小学数学问题。问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？答案：11。

## (3) 增加思维链的示例格式：

输入：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

输出：让我们一步一步地思考。罗杰一开始有 5 个球，2 罐每罐 3 个网球就是 6 个网球。 $5 + 6 = 11$ 。因此答案是 11。

示例模板：下面是一个小学数学问题。问题：{输入} 答案：{输出}

具体示例：下面是一个小学数学问题。问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？答案：让我们一步一步地思考。罗杰开始有 5 个球，2 罐每罐 3 个网球就是 6 个网球。 $5 + 6 = 11$ 。因此答案是 11。

### 例 10.5 人工标注的示例格式

- 思维链提示是一种高级提示策略，旨在增强大语言模型在各类复杂推理任务上的表现。
- 常见的推理任务包括算数推理、常识推理以及符号推理等多任务。



# 思维链生成方法

- 基于采样的方法
- 基于验证的方法
- 更加复杂的推理路径

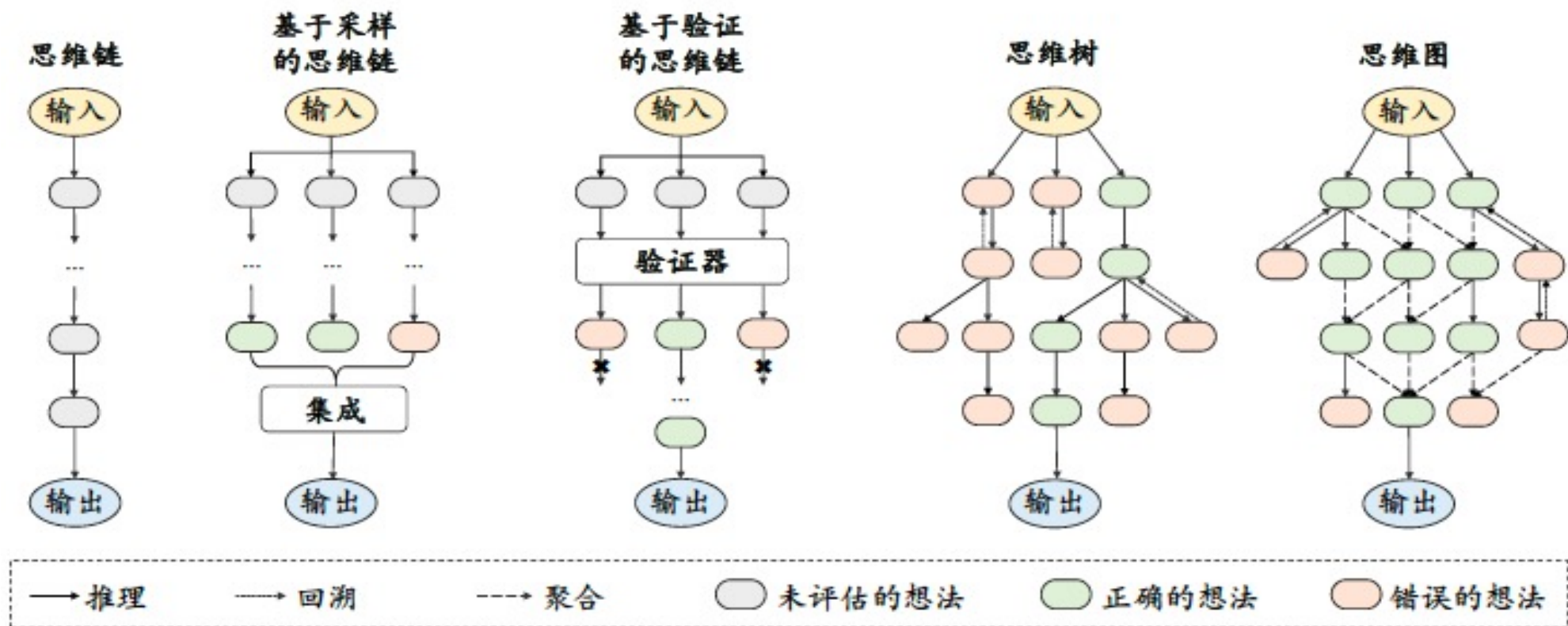
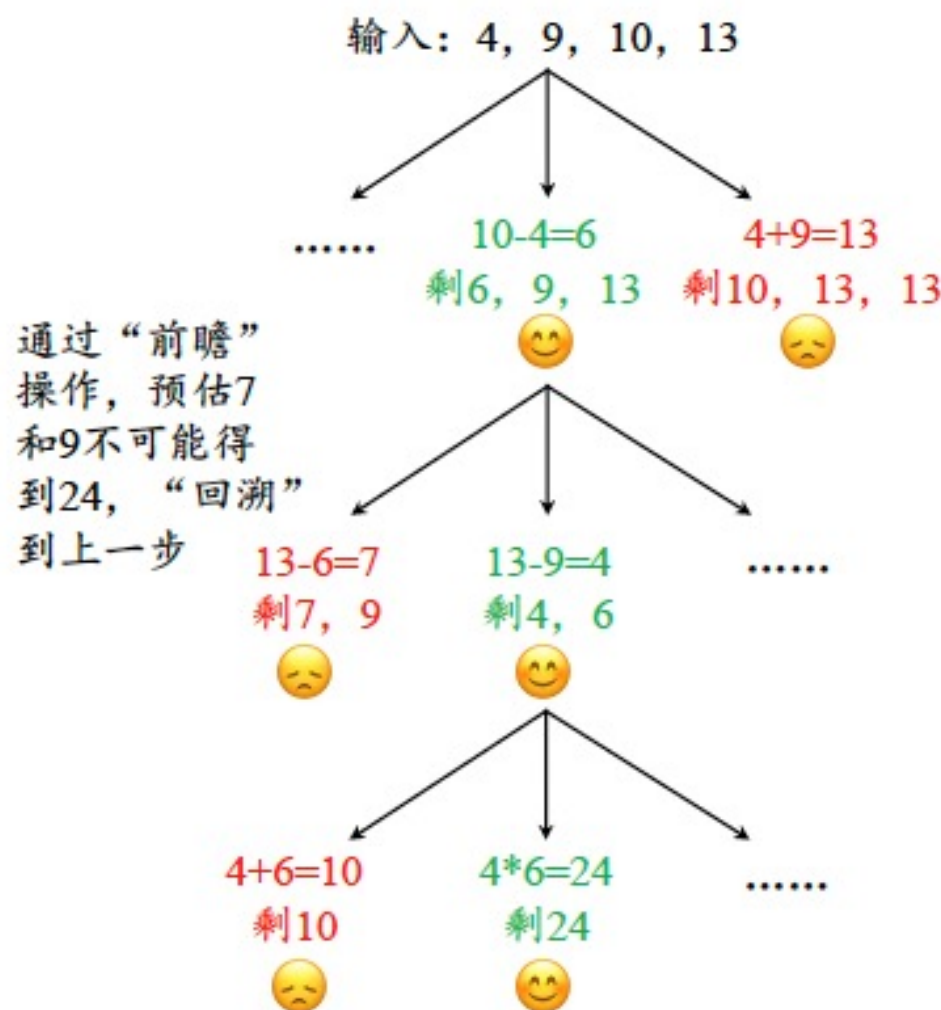


图 10.2 思维链提示技术的演化过程（图片来源：[10]）

# 思维链示例



任务: 给定四个数4、9、10、13, 如何通过加减乘除四则运算得到24?

① 生成多个可能的初始步骤, 例如 $10-4=6$ ,  $4+9=13$ .....

② 每一个步骤都会生成多个下一步骤, 例如 $10-4=6$ 之后可以生成 $13-6=7$ 或 $13-9=4$ 。

③ 对中间步骤进行打分。这里可以对当前思考步骤进行“前瞻”, 例如在当前思考步骤剩下7和9时, 能前瞻性地得知无法得到24, 应该得到一个低分。

④ 如果当前节点不太可能得到最终结果, 那么“回溯”到上一节点, 选择其他路径。例如从 $13-6=7$ 的节点回溯到父节点, 然后前进走到 $13-9=4$ 节点。

图 10.3 大语言模型使用思维树方法解决 24 点游戏

- 大语言模型的构建
- 提示学习
- 大语言模型的评测

- 常见的评测指标

表 12.1 常见评测指标分类表

评测任务	评测指标	介绍
分类任务	精确率 召回率 F1 分数	计算模型预测为正例的样本中真正为正例的比例 计算真正例的样本中被模型正确预测的比例 综合衡量模型输出的精确率和召回率
语言建模任务	困惑度	衡量模型对参考文本的建模概率
文本生成任务	BLEU ROUGE	衡量机器翻译与参考翻译之间的重叠度 衡量机器摘要对参考摘要的覆盖度
问答任务	准确率	衡量模型预测的正确答案的比例
执行类任务	成功率 Pass@ $k$	衡量模型成功完成任务的比例 估计模型生成的 $k$ 个方案中至少能通过一次的概率
偏好排序类任务	Elo 等级分	衡量模型在候选者中的相对水平



- 常见的评测范式与方法

表 12.3 评测方法及其典型评测工作

方法	评测工作	模型类型	能力/领域	数据源
基于评测基准	MMLU	基础/微调	通用	人类考试
	BIG-Bench	基础/微调	通用	人工标注
	HELM	基础/微调	通用	基准集合
	C-Eval	基础/微调	通用	人类考试
	Open LLM Leaderboard	基础/微调	通用	基准集合
基于人类评估	Chatbot Arena	微调	人类对齐	人工标注
基于模型评估	AlpacaEval	微调	指令跟随	合成
	MT-Bench	微调	人类对齐	人工标注

- 基础大语言模型的评测，即经过预训练获得的模型，通常具备丰富的世界知识与通用的语言能力。

MMLU [220] 是一个综合性的大规模评测数据集，旨在全面评估大语言模型在多个领域中的知识理解和应用能力，包括人文科学、社会科学、自然科学和工程技术等。MMLU 设置了涵盖各种领域知识的 57 种子任务。这些子任务的难度不等，既有基础知识问题，也有高级问题挑战，从而能够全面衡量模型在不同层次上的知识掌握情况。由于涵盖的知识面极为广泛，MMLU 能够有效地检测出模型在哪些领域或知识点上存在不足。例如，在理工学科领域的测试中，模型需要具有出色的数理计算和推理能力；而在社会科学领域的挑战中，模型则需要对于社会现象和理论知识具有深入的理解。

- 微调大语言模型的评测通常指特定指令或对齐需求进行微调得到的模型。
- 基于人类的评测
  - 这类评测任务通常采用开放式指令或对话形式，并邀请人类评估员对模型生成的回复进行质量评估。
- 基于模型的评测
  - 考虑到人工评测的成本高昂且耗时较长，一些研究工作使用强大的比原大语言模型（如ChatGPT和GPT4）来代替人类评估员对微调大模型的输出进行自动评分或比较。

- 人工评测

如，Chatbot Arena 项目 [68] 搭建了一个众包平台，允许用户与两个匿名的聊天大语言模型进行对话，通过根据成对的比较结果来计算不同模型的 Elo 评分。在单一评分法中，评估员则独立地对每个模型的回复进行打分，最后得到每个模型的平均得分。例如，HELM 综合评测体系 [321] 让评估员对摘要和虚假信息任务进行直接打分。

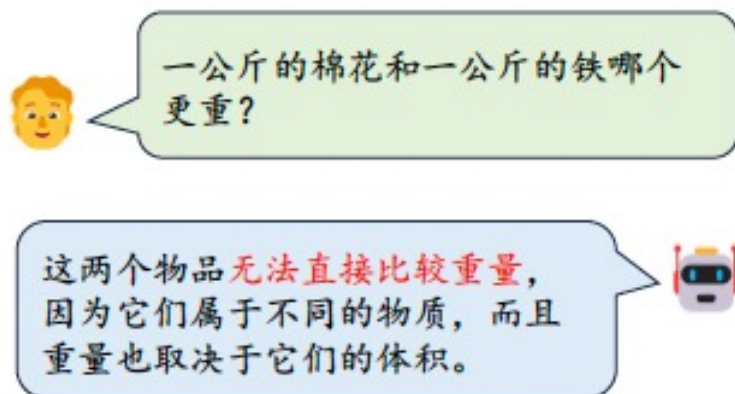


- 模型做评分

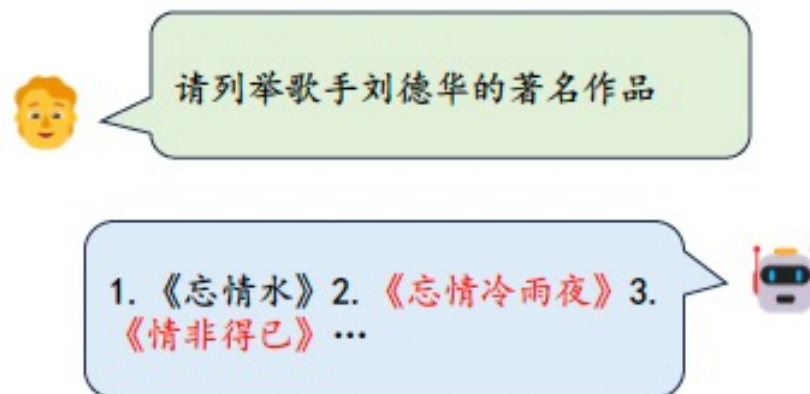
微调大模型的输出进行自动评分或比较。例如，AlpacaEval 排行榜 [315] 基于由大语言模型合成的人类需求指令作为评测任务，然后收集待评估大模型的回应，并采用 GPT-4 等大语言模型作为评测员，将待评估大语言模型的输出与参考输出进行成对比较。此外，MT-Bench 项目 [68] 也通过收集多轮问题来评估大语言模型的能力，并通过引入上下文学习和思维链提示等方法提高了基于大语言模型的评测方法的可靠性。值得注意的是，虽然 GPT-4 等闭源大语言模型在评估任务中表现

# 大模型的存在的主要问题

- 灾难性遗忘问题
- 幻觉问题



(a) 内部幻象



(b) 外部幻象

- 知识的时效性
- 前后不一致、推理不一致
- 数值计算