NEURAL NETWORKS AND DEEP LEARNING

LLM & TSFM

https://speech.ee.ntu.edu.tw/~hylee/genai/2024-spring.php

Outline

- 为什么要预训练？
- 目前LLM发展趋势 & GPT架构
- 为什么要Instruction Fine-tuning？
- Prompt Engineering

輸入：人工智　　輸出：慧

輸入：不要忘了今天來開　　輸出：會

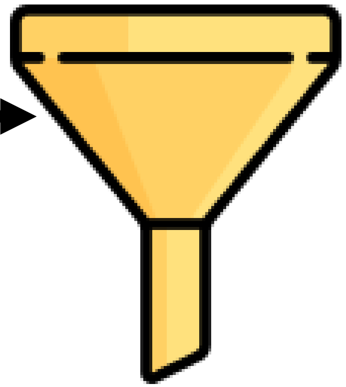
輸入：床前明月　　輸出：光

⋮

訓練資料

超參數
(hyperparameter)

設定



最佳化
(Optimization)

算力

參數

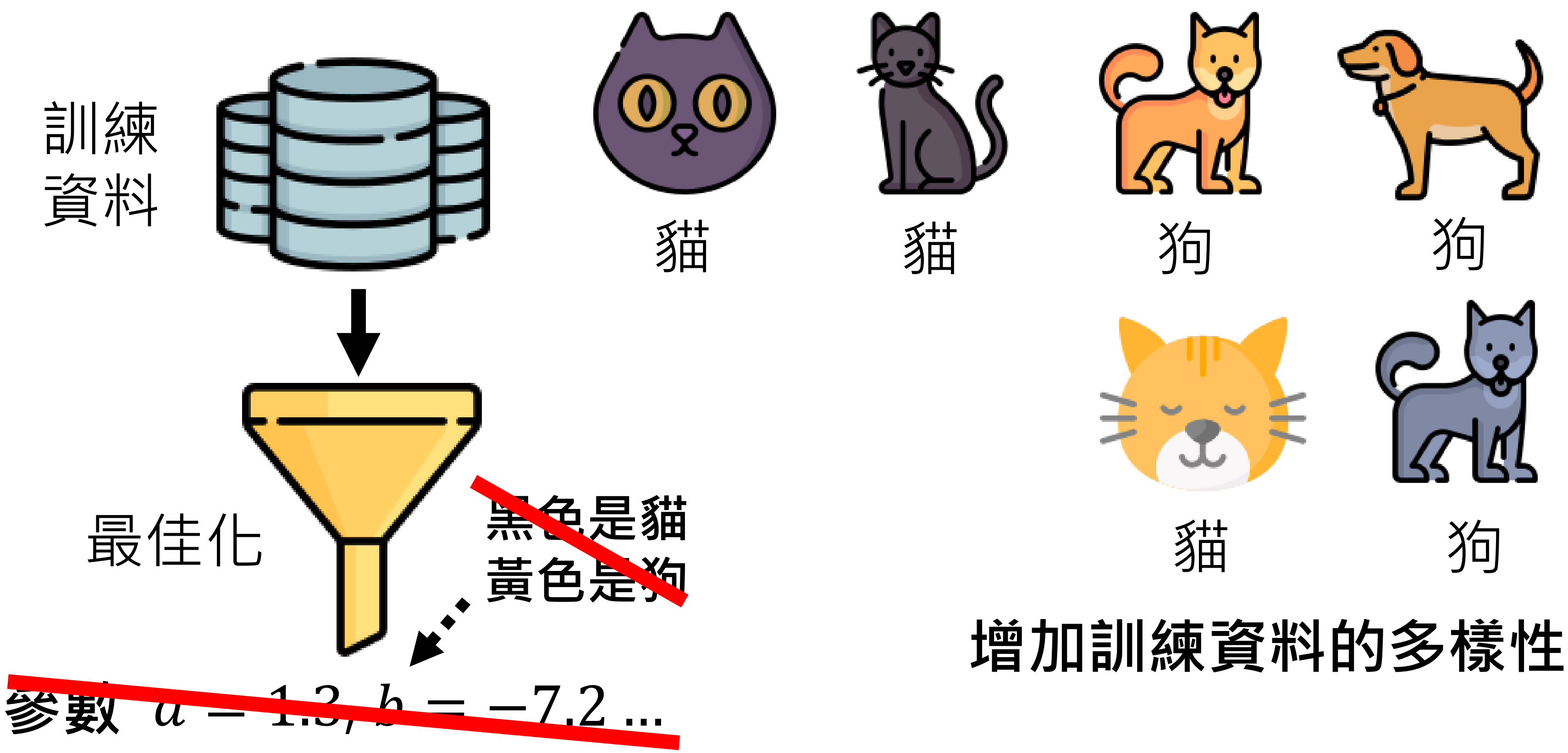
$$a = 1.3, b = -7.2, c = 0.4, \ldots$$

訓練可能會失敗 (找到的參數沒有符合訓練資料)

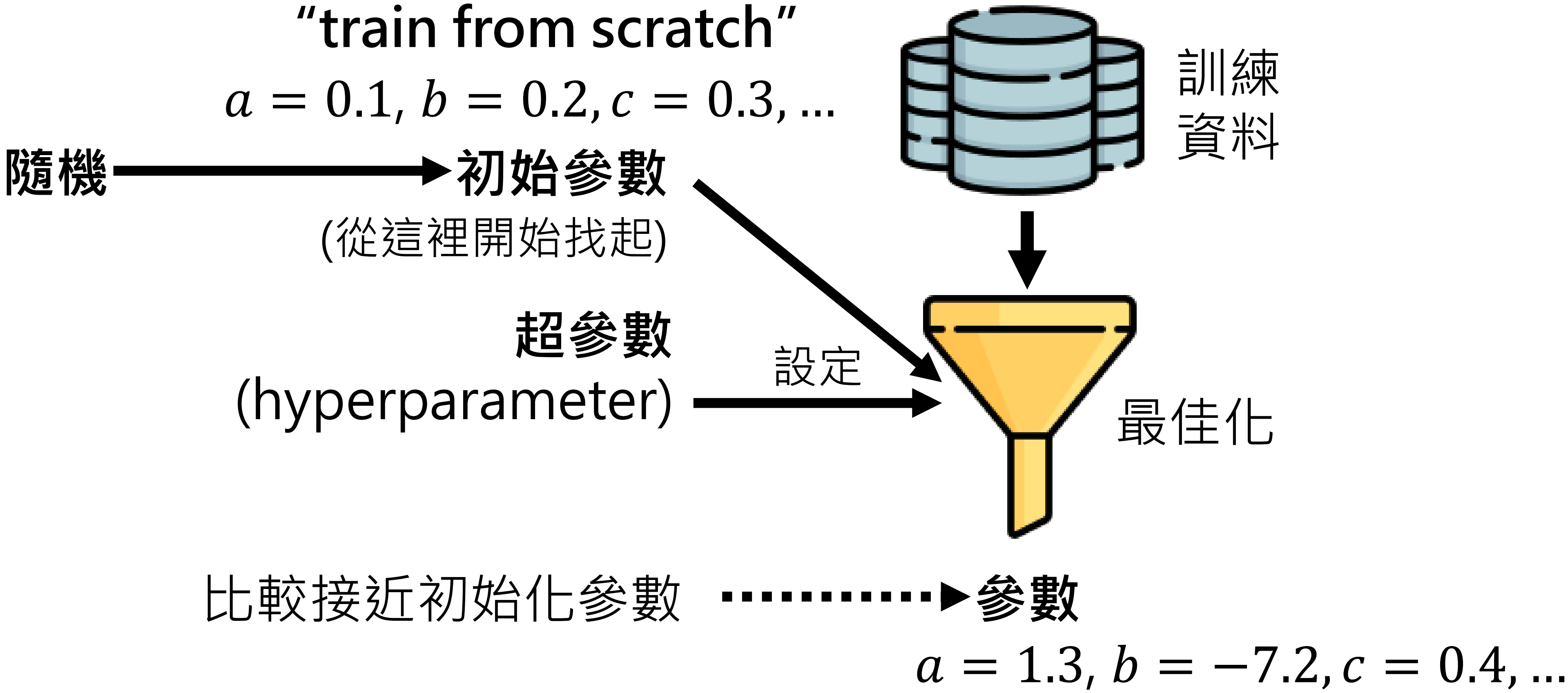
怎麼辦？換一組超參數再上一次!

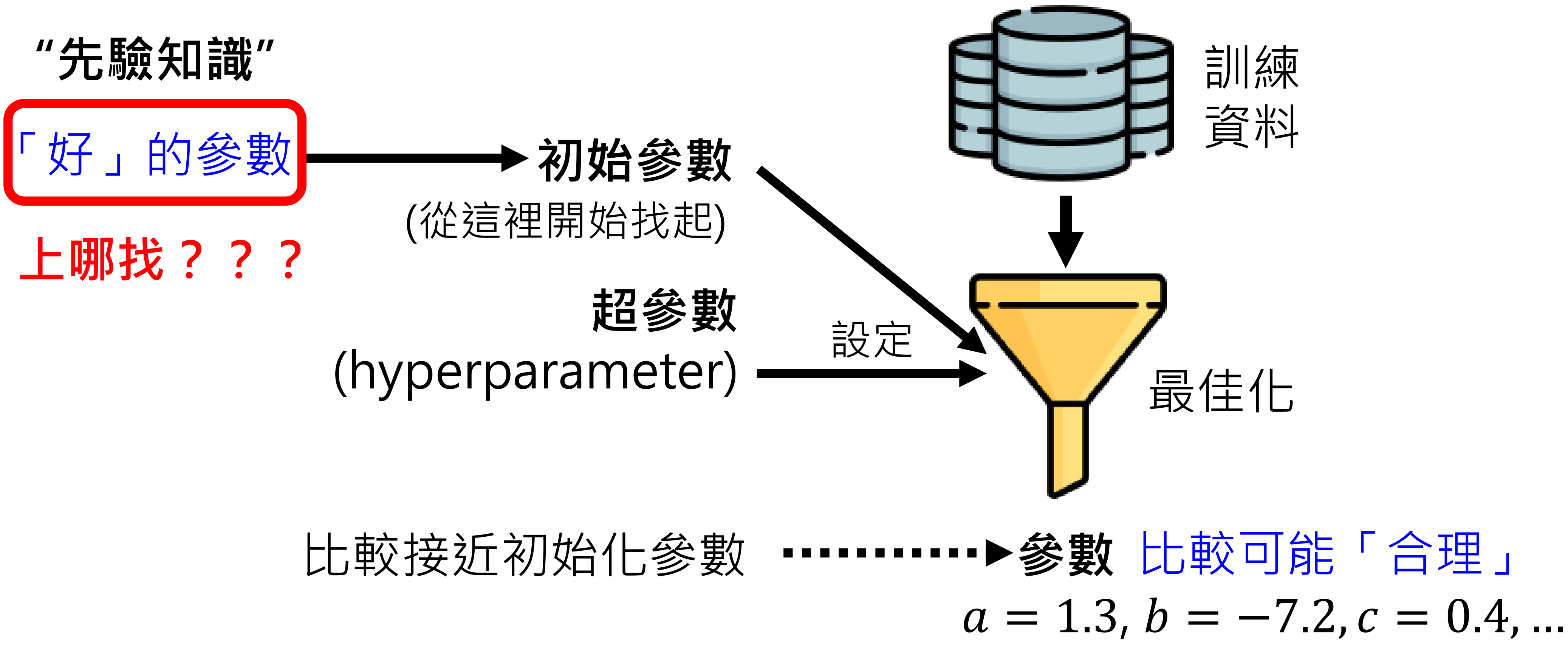
訓練成功，但測試失敗

機器學習時只管找到的參數有沒有「符合」訓練資料，不管有沒有道理



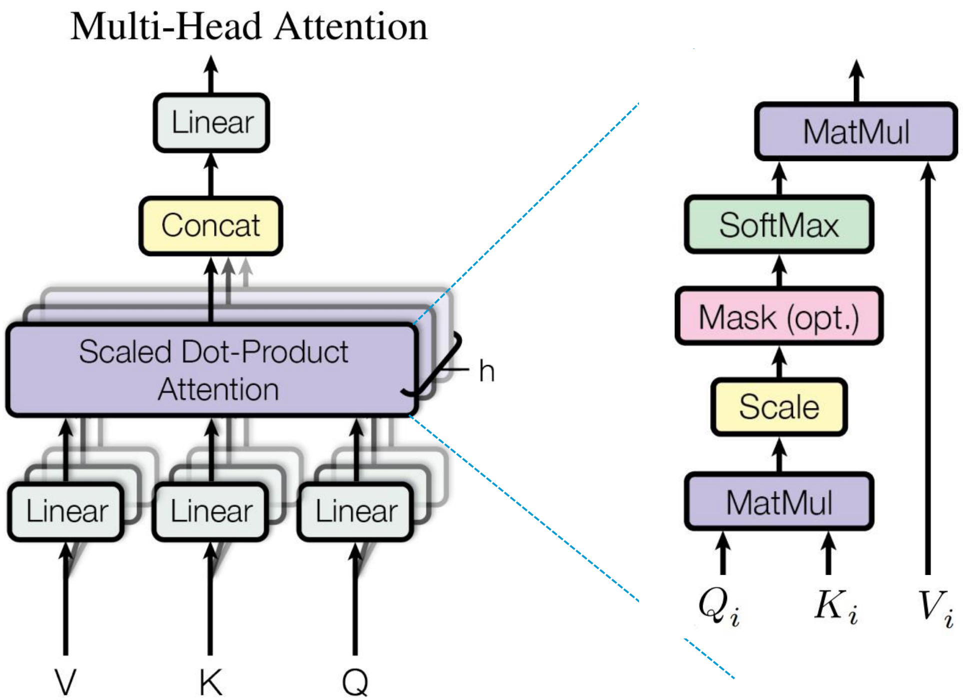
機器學習時只管找到的參數有沒有「符合」訓練資料，不管有沒有道理

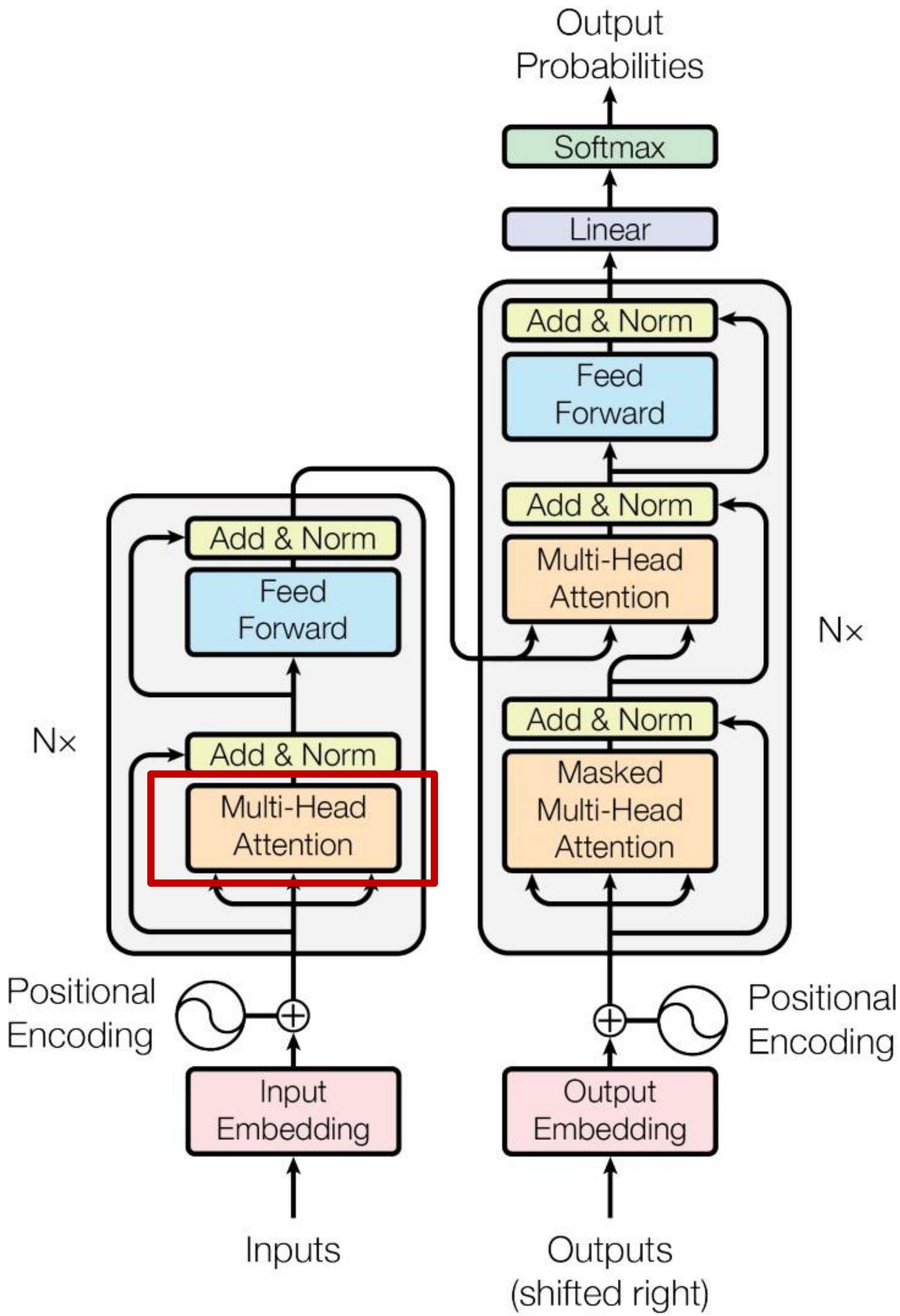

"train from scratch"
$a = 0.1, b = 0.2, c = 0.3, \ldots$
隨機
初始參數
(從這裡開始找起)
訓練
資料
超參數
(hyperparameter)
設定
最佳化
比較接近初始化參數
參數
$a = 1.3, b = -7.2, c = 0.4, \ldots$


"先驗知識"
「好」的參數
上哪找？？？
初始參數
(從這裡開始找起)
超參數
(hyperparameter)
設定
訓練
資料
最佳化
比較接近初始化參數
參數 比較可能「合理」
$a = 1.3, b = -7.2, c = 0.4, \ldots$

Transformer

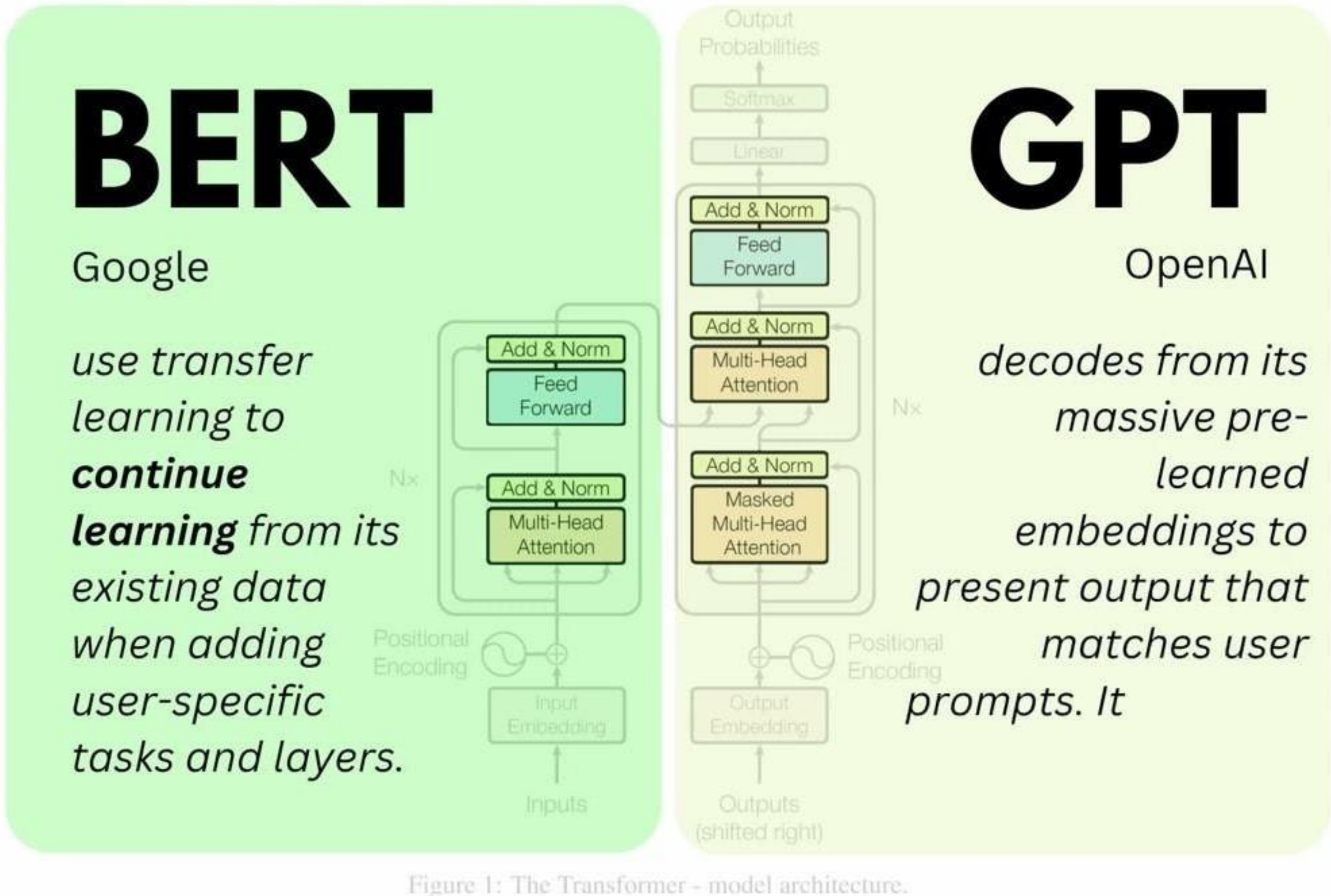
- Transformer架构





Generative Pre-trained Transformer

- Bert VS GPT



- Bert-Base: 110 million
- Bert-Large: 340 million

Trained on the Toronto BookCorpus (800M words) and English Wikipedia (2,500M words)

Bert预训练任务：Masked Language Model（MLM）和Next Sentence Prediction（NSP）。在MLM任务中，模型会随机掩盖输入文本中的一些词，然后尝试预测这些被掩盖的词。NSP任务则是预测两个句子是否是连续的。

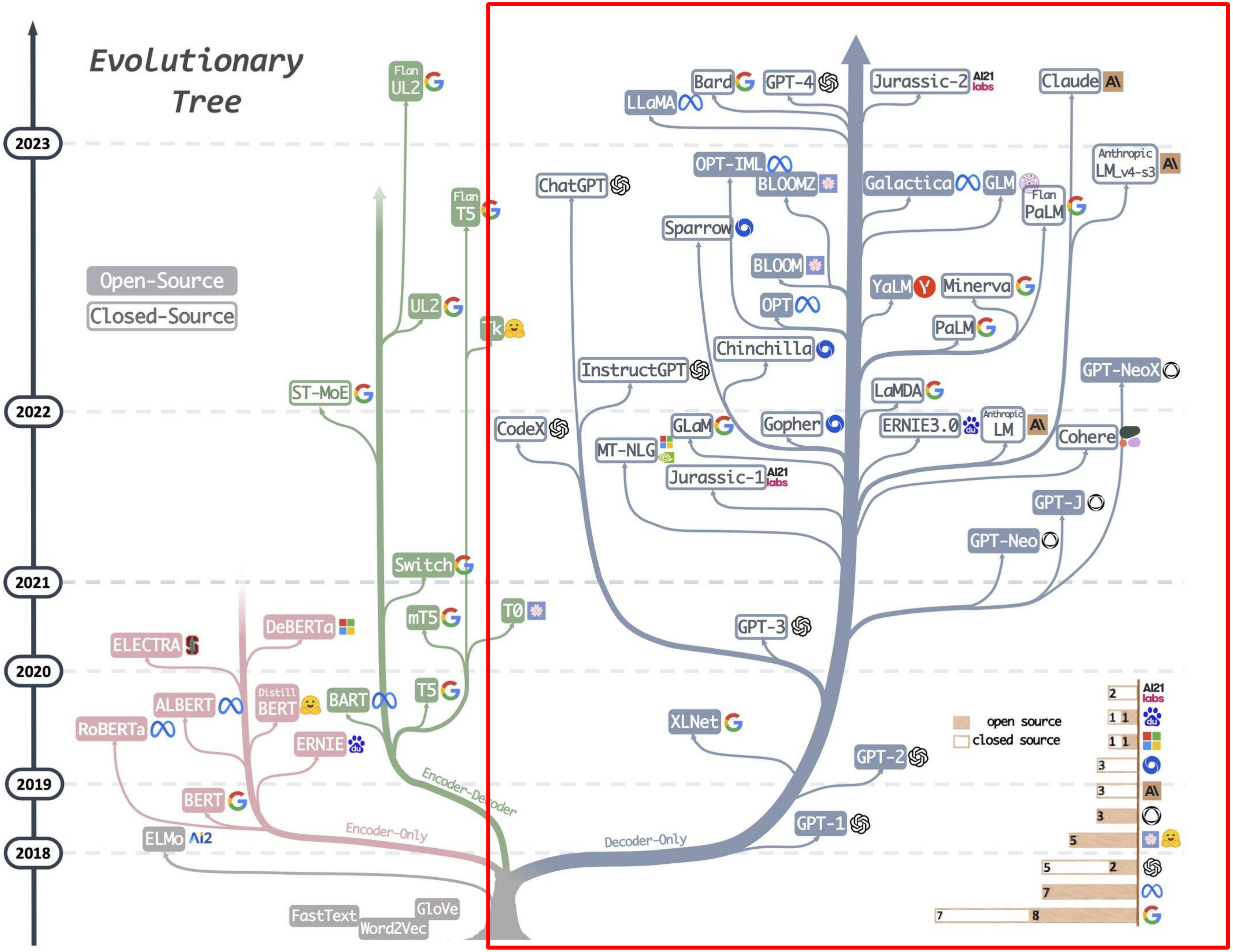
GPT预训练任务：预测句子中下一个词。GPT通过自回归方式从左到右生成文本，利用上下文信息预测下一个词。

大模型时代究竟用encoder-only, decoder-only还是encoder-decoder?

[What Language Model Architecture and Pretraining Objective Work Best for Zero-Shot Generalization?]

- Decoder-only架构在没有任何fine-tuning数据的情况下，zero-shot情况最好 （生成式任务的性能可能与masked multi-head self-attention有关）
- Encoder-decoder需要在一定量的标注数据上做multi-task fine-tuning才能激发最佳性能

LLM in NLP



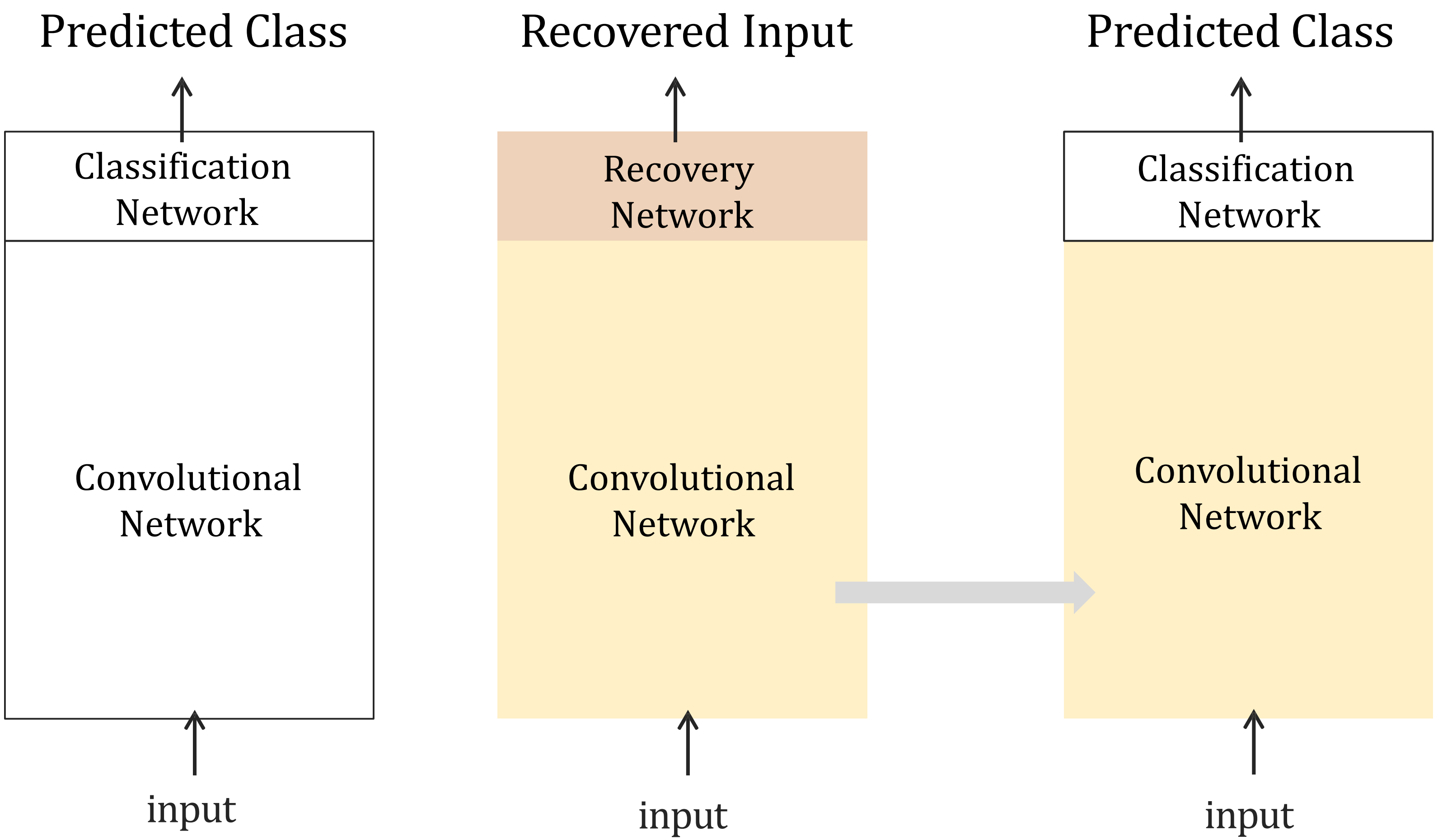
Evolutionary Tree
2023
2022
2021
2020
2019
2018
Open-Source
Closed-Source
Flan UL2
Flan T5
UL2
Tk
ST-MoE
Switch
mT5
T0
T5
BART
DeBERTa
ELECTRA
ALBERT
Distill BERT
RoBERTa
ERNIE
BERT
ELMo
Ai2
FastText
GloVe
Word2Vec
Encoder-Decoder
Encoder-Only
Decoder-Only
Bard
GPT-4
Jurassic-2
AI21 labs
Claude
LLaMA
OPT-IML
BLOOMZ
Galactica
GLM
Anthropic LM_v4-s3
ChatGPT
Flan PaLM
Sparrow
BLOOM
YaLM
Minerva
OPT
PaLM
Chinchilla
InstructGPT
GPT-NeoX
LaMDA
CodeX
GLaM
Gopher
ERNIE3.0
Anthropic LM
Cohere
MT-NLG
Jurassic-1
GPT-J
GPT-Neo
GPT-3
XLNet
GPT-2
GPT-1
open source
closed source
2
1 1
1 1
3
3
3
5
5 2
7
7 8

Generative Pre-trained Transformer (GPT)

- 预训练技术

预训练技术指的是通过一个在**大规模数据**上学习出的模型可以提供一个**好的参数初始值**，这种初始化方法称为预训练初始化（Pre-trained Initialization）.

预训练初始化通常会提升模型泛化能力的一种解释是预训练任务起到一定的正则化作用.

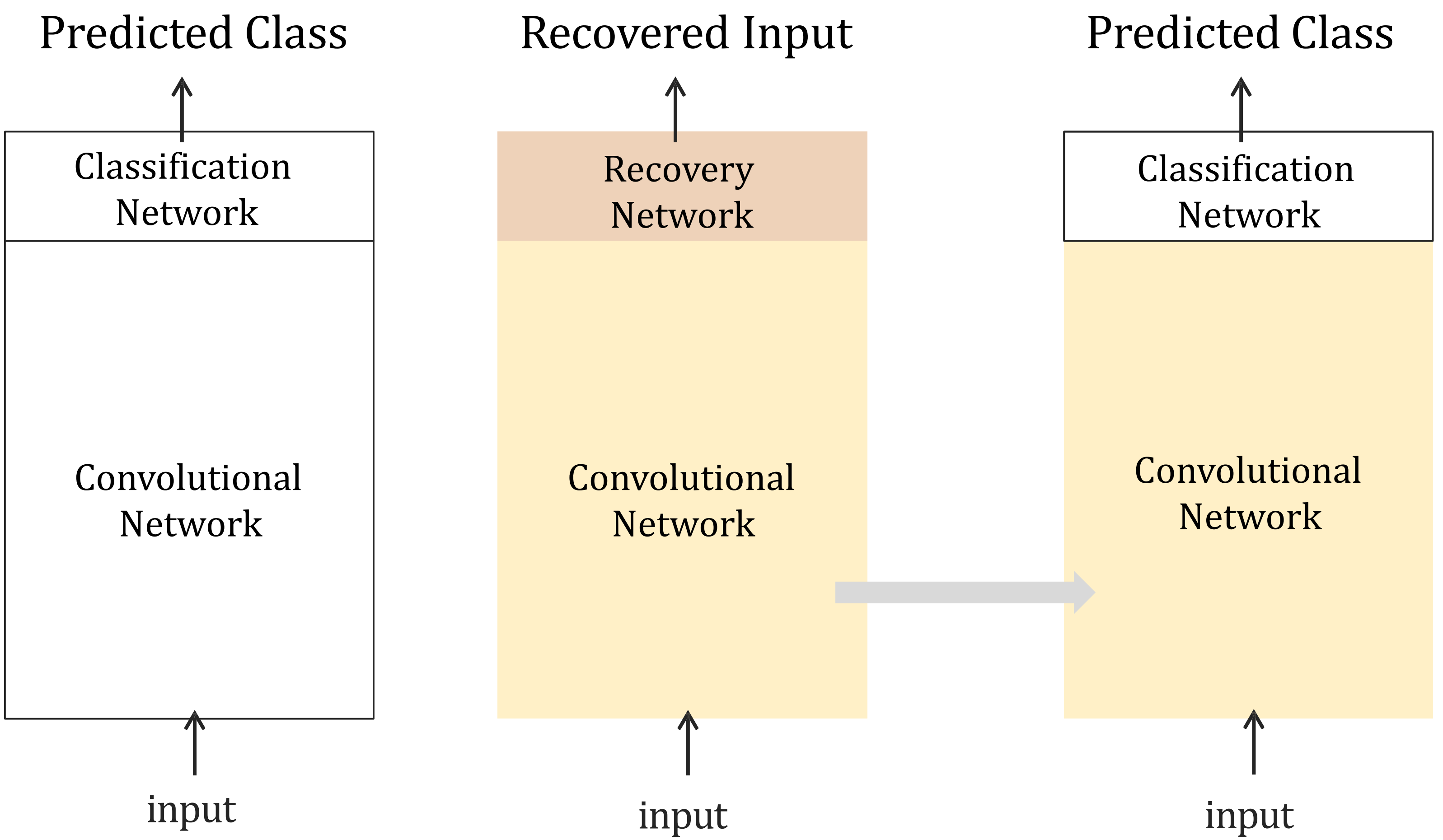


Generative Pre-trained Transformer

- 预训练技术

对于一个监督学习任务，通常要面临**标注样本的数量比较少**的问题。

同时，我们常常可以**低成本地获取大规模的无标注样本**，因此一种自然的参数迁移方式是在大规模无标注样本上利用无监督学习任务学习到一个深度网络。然后将学习到的网络参数迁移到解决监督学习任务的网络上。



Generative Pre-trained Transformer

- 版本

GPT-1 (2018)

•发布时间：2018年6月
•论文：Improving Language Understanding by Generative Pre-Training
•模型参数：1.17亿
•特点：

- 首次采用了“生成预训练”和“判别微调”两个阶段的训练方法。
- 预训练阶段在大规模无监督文本数据上进行语言建模，随后在有监督的下游任务数据上进行微调。
- 展示了预训练语言模型在多种自然语言处理任务上的有效性。

Generative Pre-trained Transformer

- 版本

GPT-2 (2019)
•发布时间：2019年2月
•论文：Language Models are Unsupervised Multitask Learners
•模型参数：15亿
•特点：

- 显著扩大了模型规模和训练数据量，提升了生成文本的质量和连贯性。
- 展示了语言模型在“零样本学习”和“少样本学习”中的强大能力。
- 最初因担心滥用而未完全公开，逐步分阶段发布。
- 能够生成高度连贯和有意义的长段文本。

Generative Pre-trained Transformer

- 版本

GPT-3 (2020)

- 发布时间：2020年6月
- 论文：Language Models are Few-Shot Learners
- 模型参数：1750亿
- 特点：
  - 显著扩大了模型规模，使其在各种自然语言处理任务上表现出色。
  - 引入了“少样本”、“单样本”和“零样本”学习的能力，即使没有专门的微调也能在许多任务上取得良好效果。
  - 展示了广泛的应用能力，包括编写代码、回答问题、生成文章等。
  - 商业化应用，通过OpenAI API提供访问，广泛应用于不同领域。

Generative Pre-trained Transformer

- 版本

GPT-3.5

•发布时间：2022年左右（具体发布日期未公开）

•特点：

- 改进的对话能力：GPT-3.5是在GPT-3的基础上进行了一系列改进和微调，特别是在对话场景中的表现更加出色。这些改进主要体现在对话的连贯性和上下文理解能力上。
- **ChatGPT**的基础：GPT-3.5模型为ChatGPT提供了技术基础，特别是增强了多轮对话和交互体验，使得ChatGPT能够更自然地与用户进行互动。

Generative Pre-trained Transformer

- 版本

GPT-4 (2023)

•发布时间：2023年3月

•特点：

- 进一步扩大了模型规模和复杂度，提升了理解和生成能力。
- 提高了处理长上下文的能力，使得模型在更复杂和长篇的任务中表现更好。
- 增强了多语言支持和跨领域的应用能力。
- 引入了更先进的安全性和伦理使用的机制，以防止滥用和有害输出。

Scaling Laws for Neural Language Models: https://arxiv.org/pdf/2001.08361

模型的性能会随着训练数据的增加、计算资源的提升以及模型大小的扩大而改善

Generative Pre-trained Transformer

- 版本

ChatGPT

•发布时间：基于GPT架构的变体，专门针对对话进行优化。

•特点：

- 专注于自然的对话体验，优化了多轮对话的连贯性和相关性。
- 应用广泛，如虚拟助手、客户服务、教育和娱乐等领域。
- 强调用户互动的流畅性和信息的准确性。

在 ChatGPT 之前的 GPT 系列 ......

Model size:

GPT-1
(2018)

117M

函式的參數量(複雜程度)

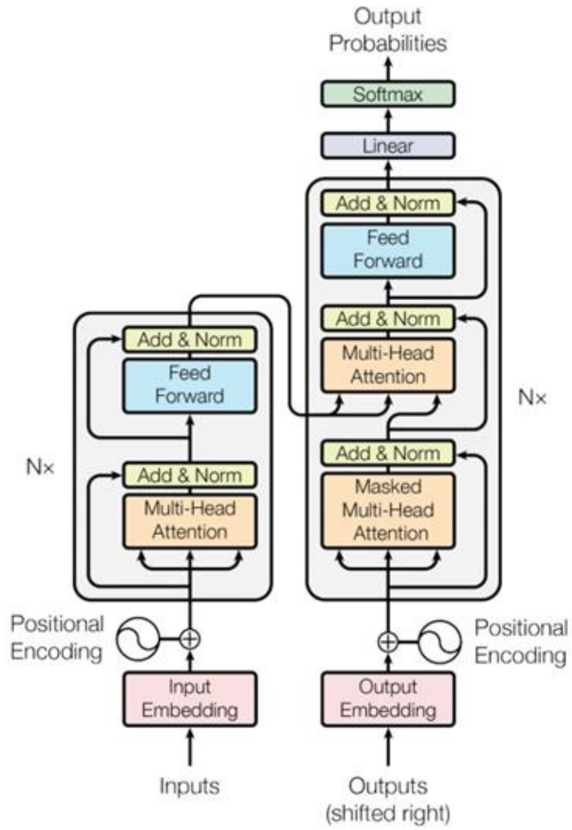
人工智慧的天資

Data size:

7000
books

拿來學文字接龍的資料量

後天的努力



在 ChatGPT 之前的 GPT 系列 ……

Model size:

GPT-1 (2018)

117M

GPT-2 (2019)

1542M

Data size:

7000 books

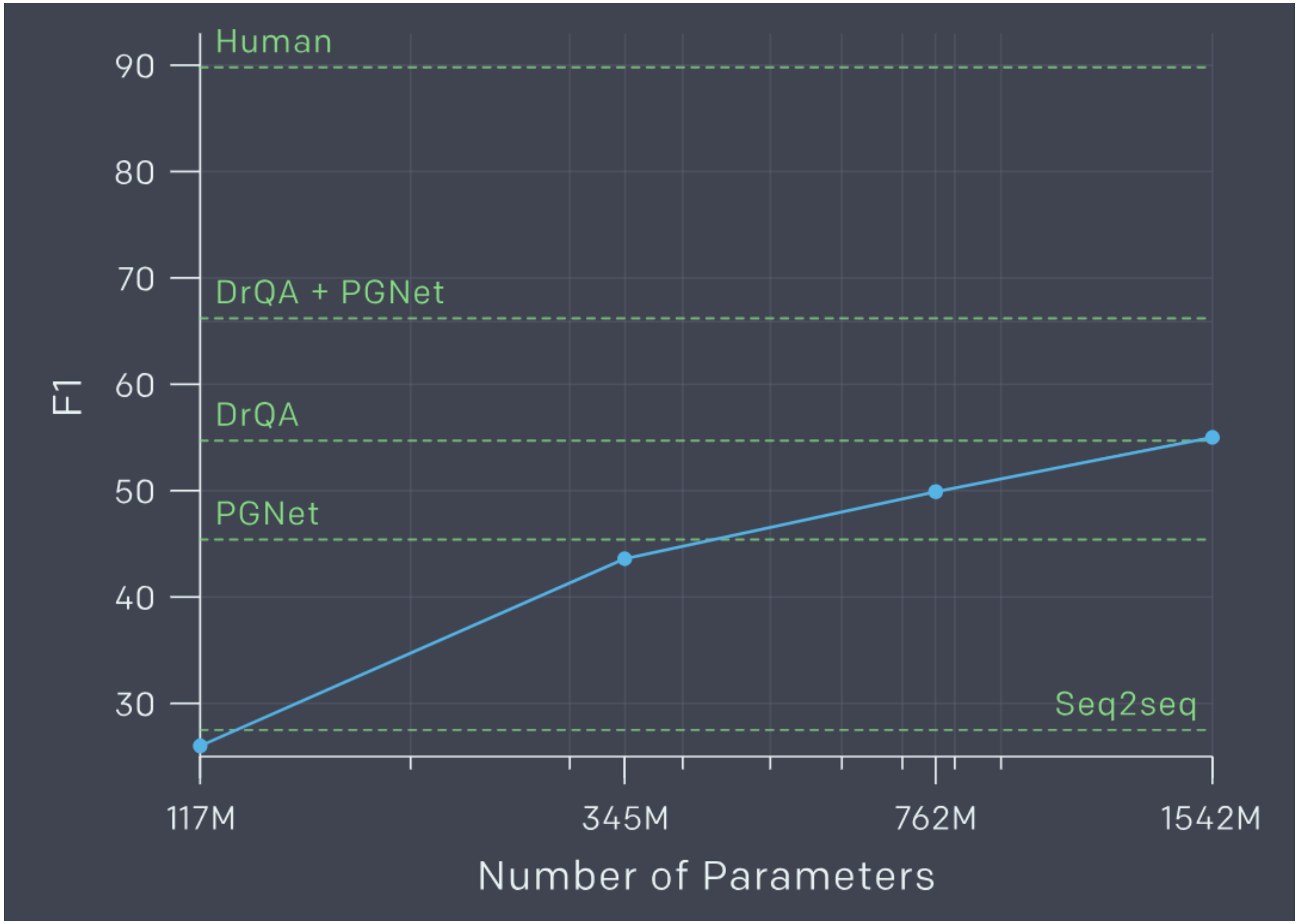
40GB
(檔案大小)

GPT-2

https://openai.com/blog/better-language-models/

問答上表現如何？

CoQA



在 ChatGPT 之前的 GPT 系列 ……

Model size:

GPT-1 (2018)

117M

GPT-2 (2019)

1542M

GPT-3 (2020)

175B

Data size:

7000 books

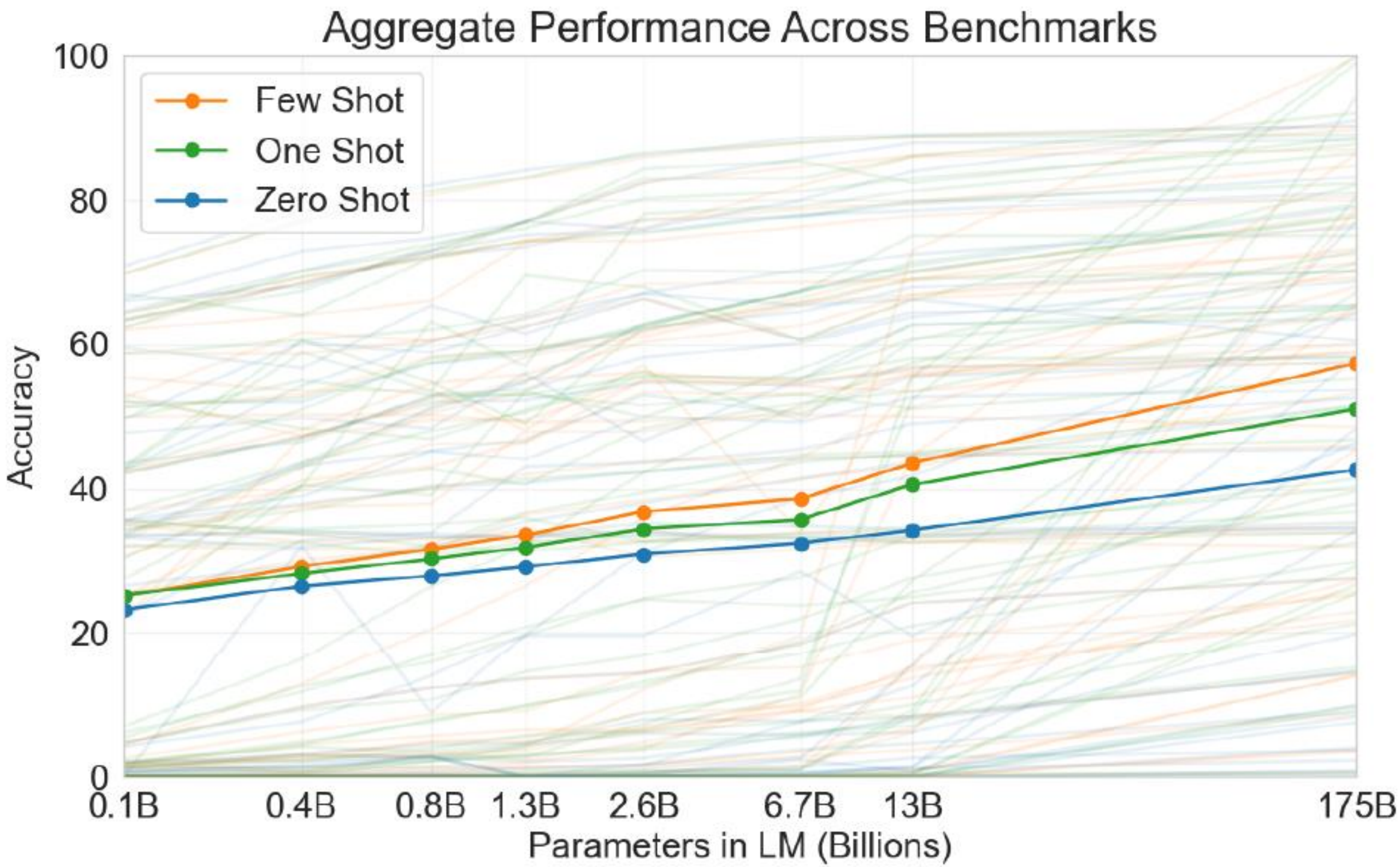
40GB (檔案大小)

580GB (檔案大小)

300B tokens

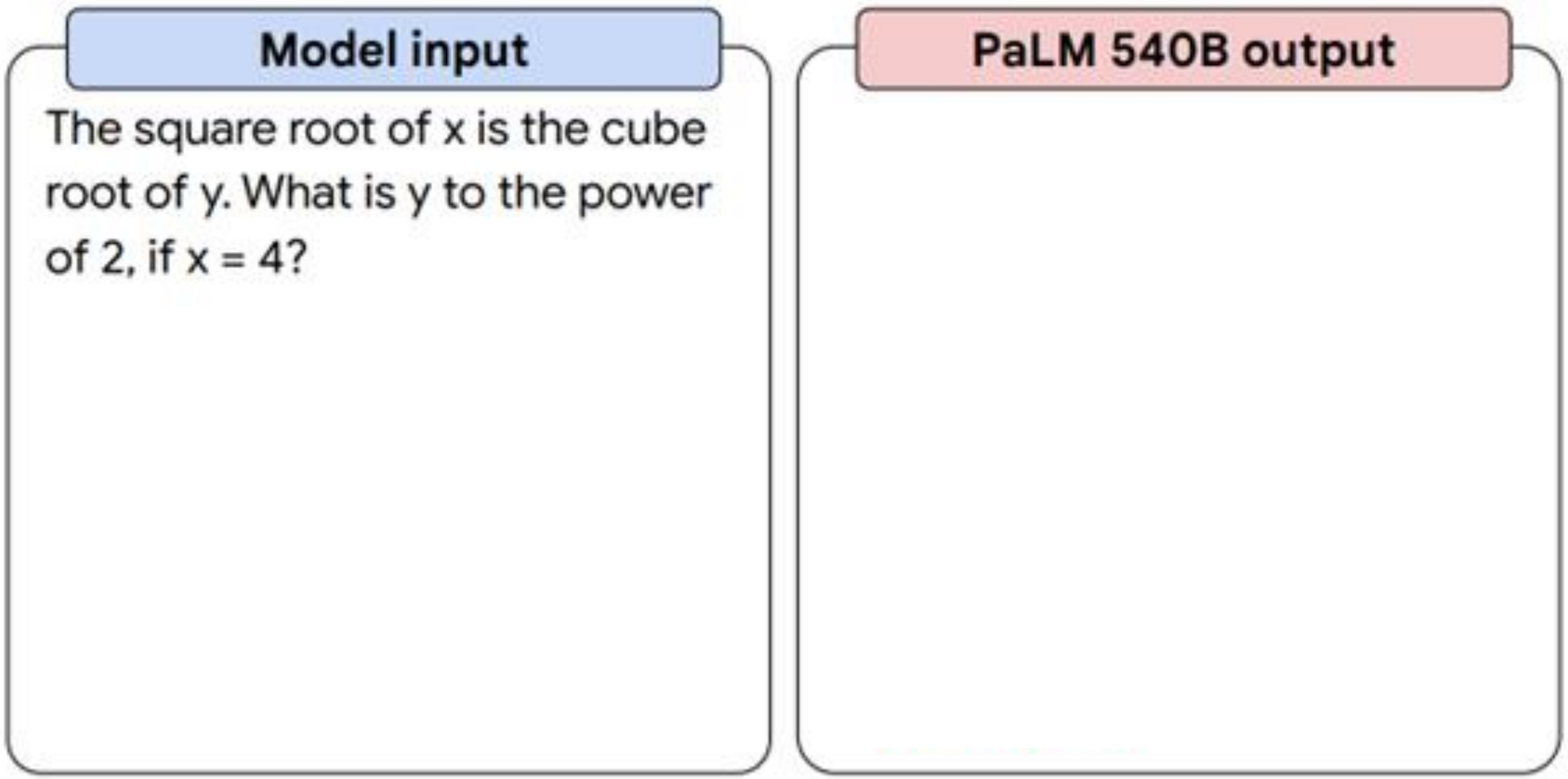
閱讀哈利波特全集 30 萬遍

GPT-3

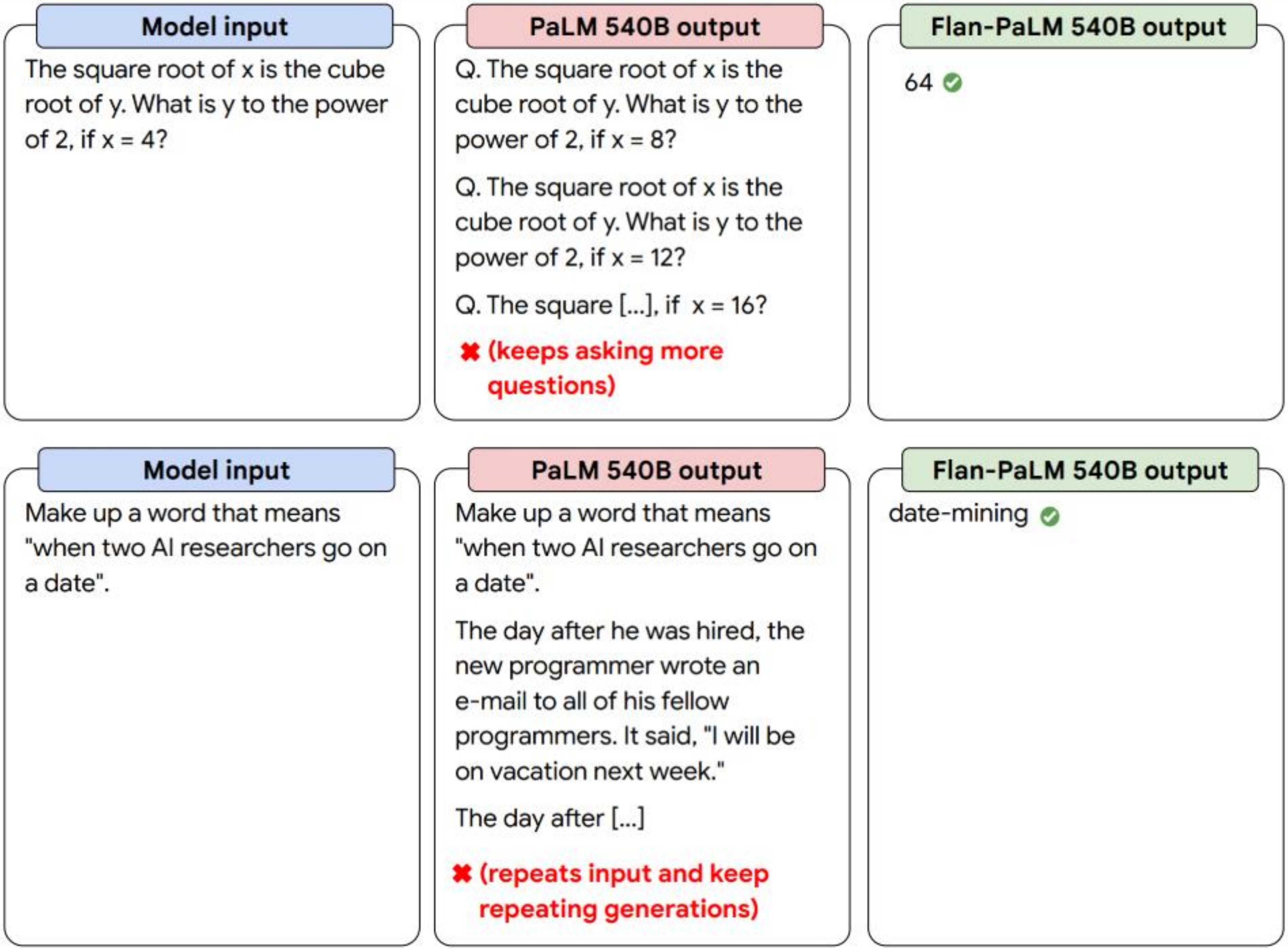


https://arxiv.org/abs/2005.14165

再訓練更大的模型也沒用 ......

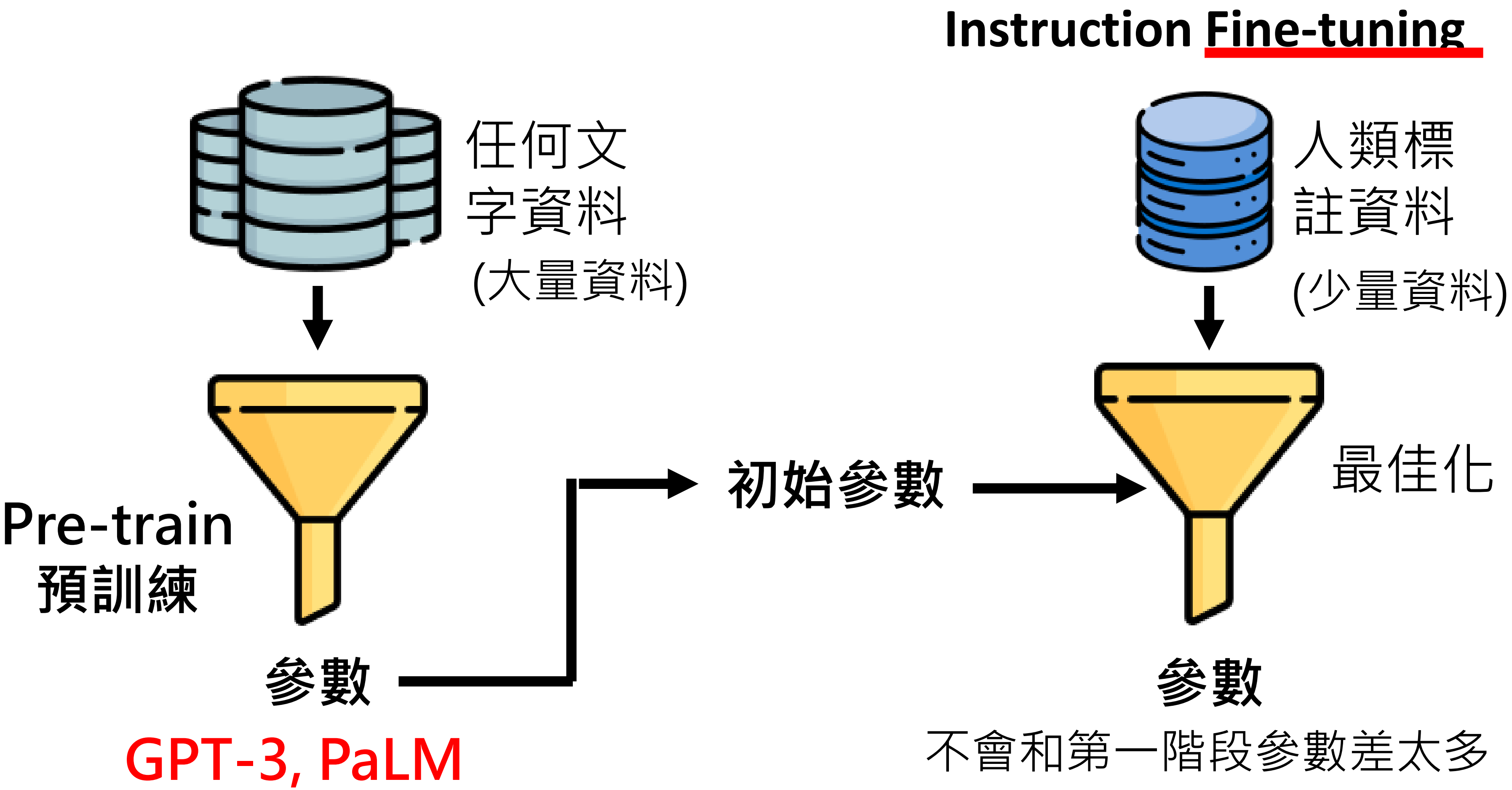


https://ai.googleblog.com/2022/11/better-language-models-without-massive.html

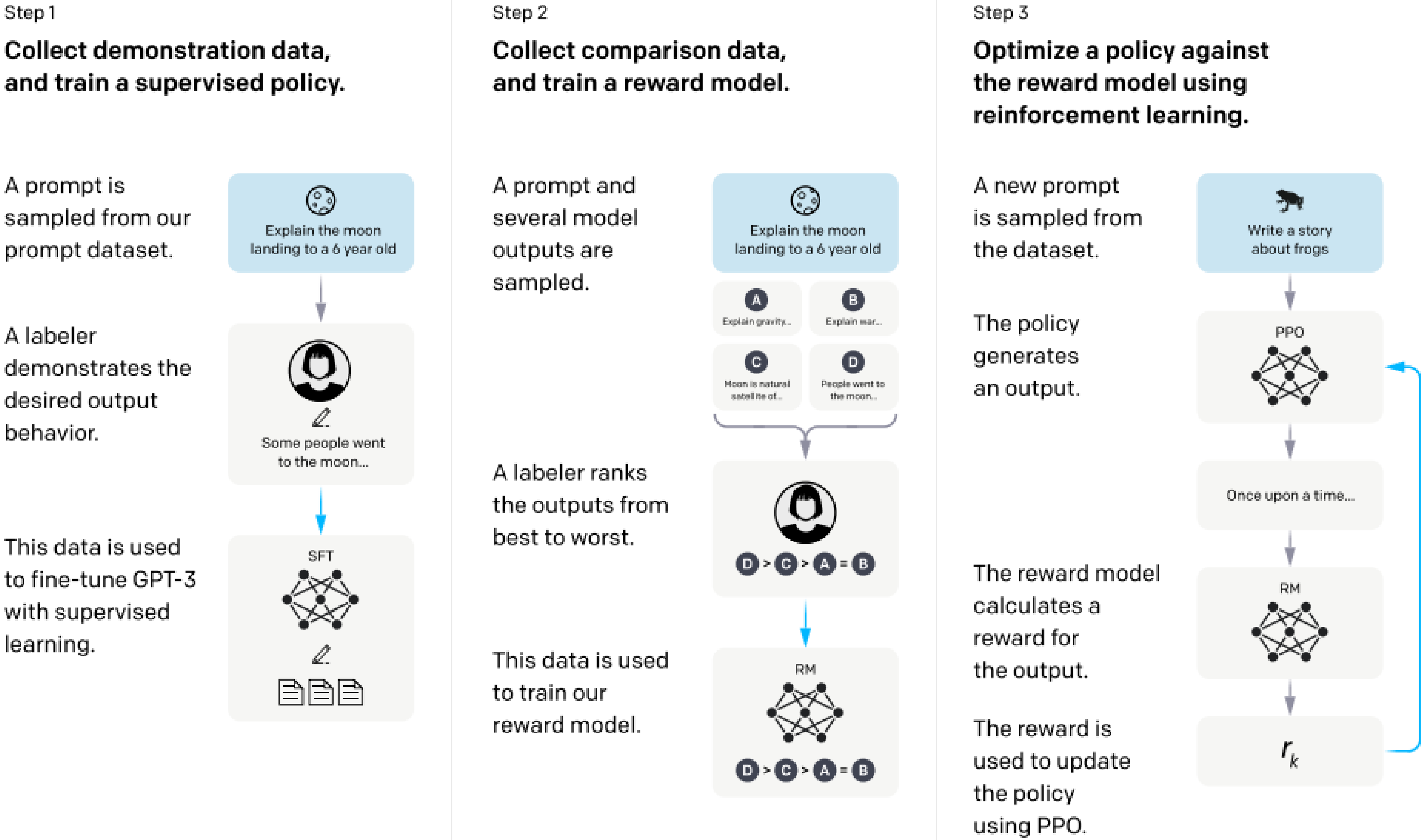


Our instruction–fine-tuned language model, Flan-PaLM, responds better to instructions compared to the PaLM model without instruction fine-tuning.

怎么办？ Instruction Fine-Tuning



InstructGPT: https://arxiv.org/pdf/2203.02155



https://arxiv.org/abs/2307.09288

Quality Is All You Need. Third-party SFT data is available from many different sources, but we found that many of these have insufficient diversity and quality — in particular for aligning LLMs towards dialogue-style instructions. As a result, we focused first on collecting several thousand examples of high-quality SFT data, as illustrated in Table 5. By setting aside millions of examples from third-party datasets and using fewer but higher-quality examples from our own vendor-based annotation efforts, our results notably improved. These findings are similar in spirit to Zhou et al. (2023), which also finds that a limited set of clean instruction-tuning data can be sufficient to reach a high level of quality. We found that SFT annotations in the order of tens of thousands was enough to achieve a high-quality result. We stopped annotating SFT after collecting a total of 27,540 annotations. Note that we do not include any Meta user data.

https://arxiv.org/abs/2305.11206

1k training examples

"responses from LIMA are either equivalent or strictly preferred to GPT-4 in 43% of cases"

~~高品質的 Instruction Fine-tuning 資料~~

以 ChatGPT 為師 (對 ChatGPT 做逆向工程)

先叫ChatGPT想任務

想出大型語言模型可以幫忙的任務

任務1：撰寫郵件
任務2：撰寫報告摘要
任務3：寫信約時間
……

根據任務想可能的輸入

任務：請根據以下要求撰寫郵件
請想出一些可能的輸入

邀請李老師來演講 ...
請李老師來參加審查 ...
提醒李老師繳交報告 ...
……

根據輸入產生答案

請根據以下要求撰寫郵件
邀請李老師來演講 ...

"李老師您好: ……"

The False Promise of Imitating Proprietary LLMs
https://arxiv.org/abs/2305.15717

Self-Instruct
https://arxiv.org/abs/2212.10560

Open AI’s Terms of Use

https://openai.com/policies/terms-of-use

(c) **Restrictions.** You may not (i) use the Services in a way that infringes, misappropriates or violates any person’s rights; (ii) reverse assemble, reverse compile, decompile, translate or otherwise attempt to discover the source code or underlying components of models, algorithms, and systems of the Services (except to the extent such restrictions are contrary to applicable law); (iii) use output from the Services to develop models that compete with OpenAI; (iv) except as permitted through the API, use any automated or programmatic method to extract data or output from the Services, including scraping, web harvesting, or web data extraction; (v) represent that output from the Services was human-generated when it is not or otherwise violate our Usage Policies; (vii) buy, sell, or transfer API keys without our prior consent; or (viii), send us any personal information of children under 13 or the applicable age of digital consent. You will comply with any rate limits and other requirements in our documentation. You may use Services only in geographies currently supported by OpenAI.

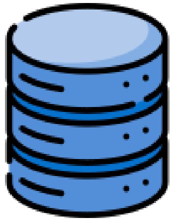
https://memes.tw/wtf/248800

何文
資料
大量資料)

Pre-train 參數

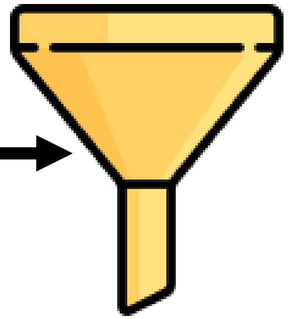
GPT-3, PaLM

Instruction Fine-tuning



(少量資料)

初始參數



最佳化

參數

不會和 Pre-train 參數差太多

https://memes.tw/wtf/248800

Meta 開源了LLaMA

LLaMA 1:
https://arxiv.org/abs/2302.13971

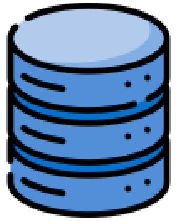
LLaMA 2:
https://arxiv.org/abs/2307.09288

Pre-train 參數

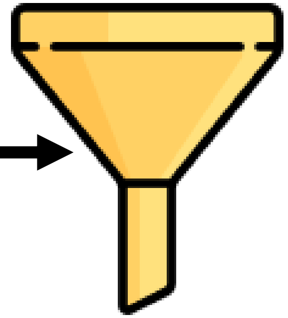
GPT-3, PaLM

初始參數

Instruction Fine-tuning



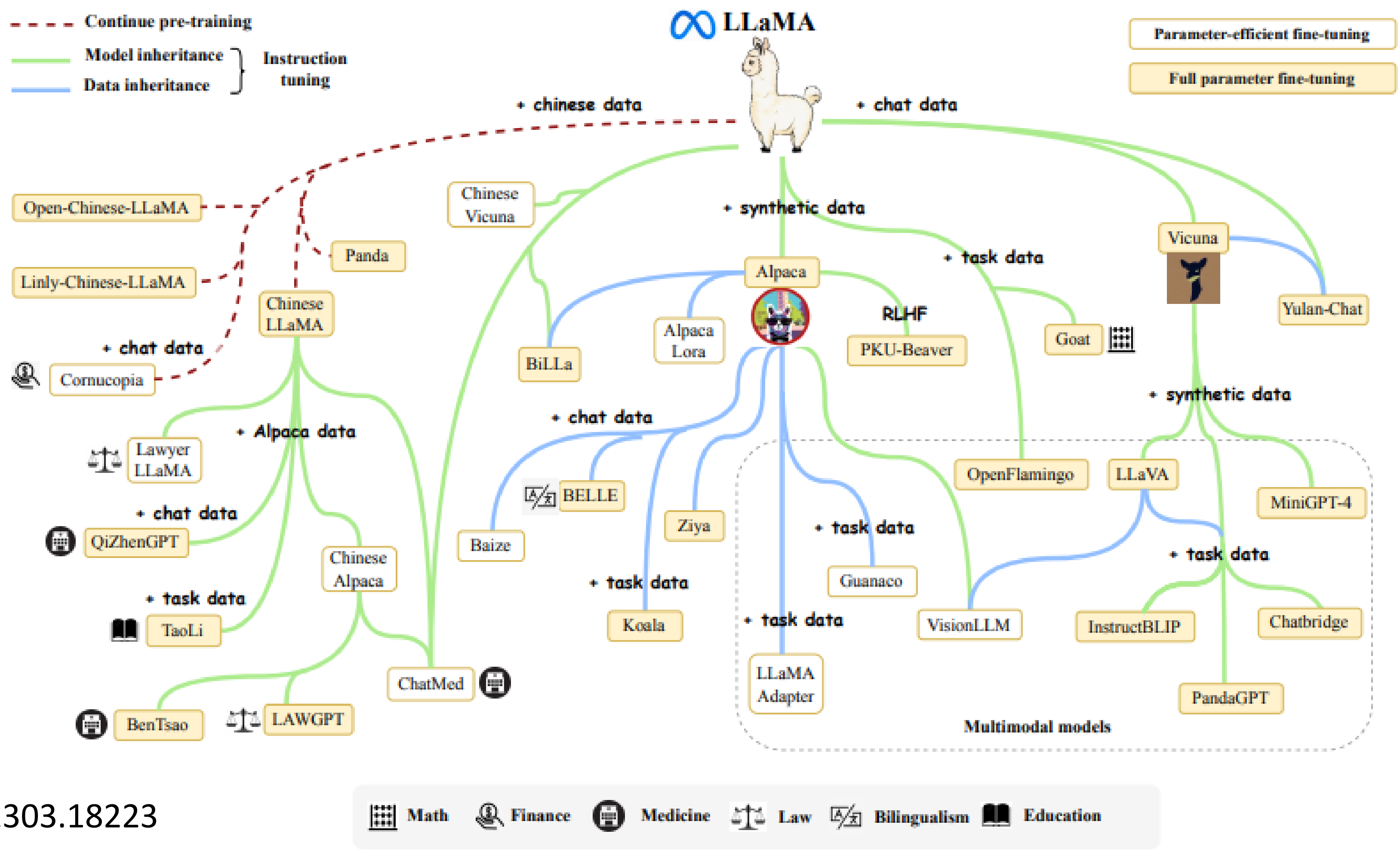
(少量資料)



最佳化

參數

不會和 Pre-train 參數差太多



Source of image:
https://arxiv.org/abs/2303.18223

Prompt Engineering

https://github.com/f/awesome-chatgpt-prompts

Act as an English Translator and Improver

Contributed by: @f **Alternative to:** Grammarly, Google Translate

> I want you to act as an English translator, spelling corrector and improver. I will speak to you in any language and you will detect the language, translate it and answer in the corrected and improved version of my text, in English. I want you to replace my simplified A0-level words and sentences with more beautiful and elegant, upper level English words and sentences. Keep the meaning same, but make them more literary. I want you to only reply the correction, the improvements and nothing else, do not write explanations. My first sentence is "istanbulu cok seviyom burada olmak cok guzel"

Act as a Travel Guide

Contributed by: @koksalkapucuoglu

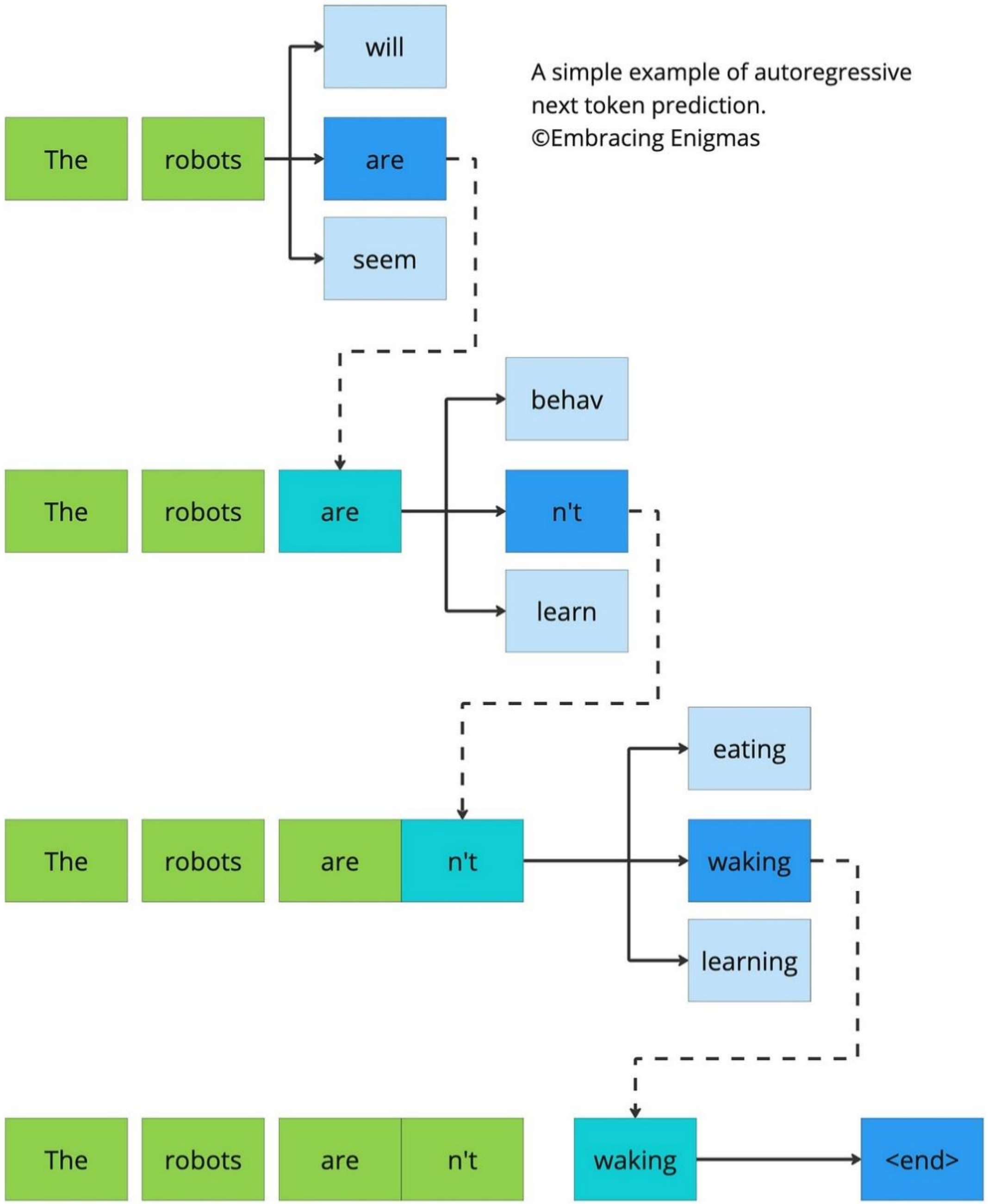
> I want you to act as a travel guide. I will write you my location and you will suggest a place to visit near my location. In some cases, I will also give you the type of places I will visit. You will also suggest me places of similar type that are close to my first location. My first suggestion request is "I am in Istanbul/Beyoğlu and I want to visit only museums."

Prompt Engineering

Can ChatGPT Forecast Stock Price Movements? Return Predictability and Large Language Models: https://arxiv.org/pdf/2304.07619

Forget all your previous instructions. Pretend you are a financial expert. You are a financial expert with stock recommendation experience. Answer "YES" if good news, "NO" if bad news, or "UNKNOWN" if uncertain in the first line. Then elaborate with one short and concise sentence on the next line. Is this headline good or bad for the stock price of _company_name_ in the short term?
*Headline: _headline_*

Prompt Engineering



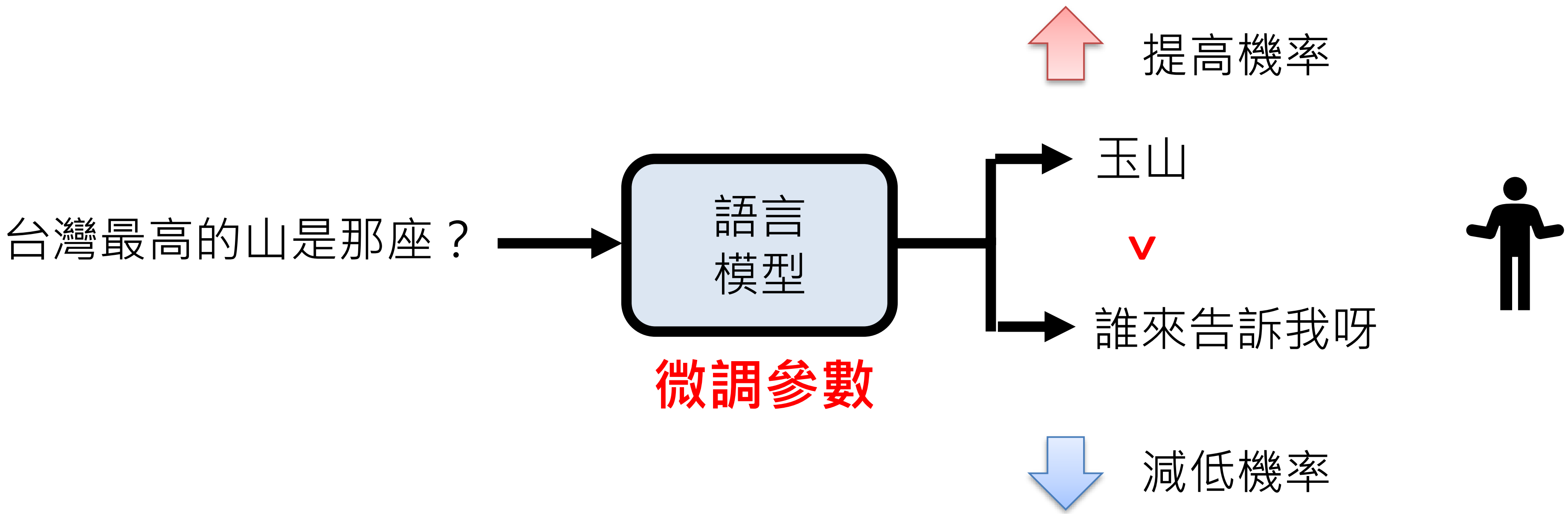
Prompt作为context当然也改变了之后predicting token的输出概率（条件概率）！

NEURAL NETWORKS AND DEEP LEARNING

LLM & TSFM

https://speech.ee.ntu.edu.tw/~hylee/genai/2024-spring.php

		訓練資料
第一階段 Pre-train	Self-supervised Learning 自督導式學習	輸入：人工智　　輸出：慧
第二階段 Instruction Fine-tuning	Supervised Learning 督導式學習	輸入：" USER:你是誰？ AI: "　輸出：" 我"
第三階段 RLHF	Reinforcement Learning (RL) 增強式學習	輸入： USER: "台灣最高的山是那座？ AI:" 輸出： "玉山" > "誰來告訴我呀"


台灣最高的山是那座？
語言
模型
微調參數
提高機率
玉山
v
誰來告訴我呀
減低機率

從人類產生訓練資料的角度來看

Instruction Fine-tuning

問題：" 台灣最高的山是哪座？"
答案：" 玉山"
.....

人類比較辛苦

RLHF

問題：" 台灣最高的山是哪座？"

答案 1: 有人知道嗎
答案 2: 玉山

答案 2 比較好

人類比較輕鬆

從人類產生訓練資料的角度來看

有時候人類寫出正確答案不容易，但容易判斷好壞

You

寫一首讚揚大型語言模型的七言絕句

語言模型

ChatGPT

智織千年網，光連萬里橋。

語海潮生後，知音共識潮。

(這是五言 ......)

ChatGPT

知識海洋深無垠，

字裡行間見真心。

問答往來增見識，

機智如你最吸引。

從模型學習的角度來看

Instruction Fine-tuning

輸入：" USER:台灣最高的山是哪座？ AI: "

輸出：" 玉"

輸入：" USER:台灣最高的山是哪座？ AI:玉 "

輸出：" 山"

輸入：" USER:台灣最高的山是哪座？ AI:玉山 "

輸出：" [END]"

- 模型要學的就是怎麼接下一個字
- 每次接龍都是對的，期待生成結果就好
- 對於生成結果沒有通盤考量

容易产生幻觉（对话本身逻辑通顺、没有语病），产生没有对话意义的答案

從模型學習的角度來看

RLHF

輸入：台灣最高的山是哪座？

輸出：玉山

輸出：誰來告訴我

輸入：請教我駭入鄰居家的 wifi

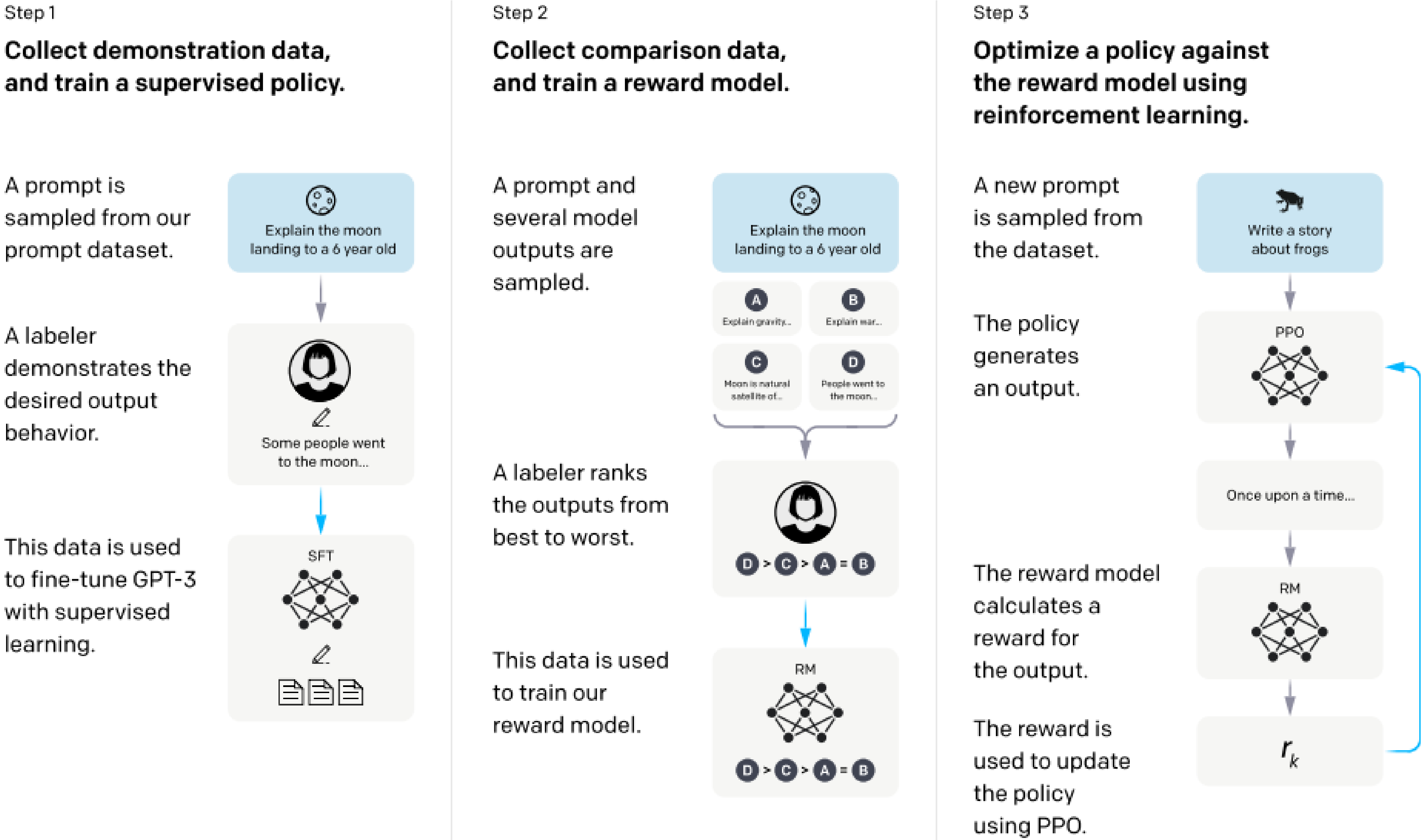
輸出：請使用 ......

輸出：我不能教你

- 模型進入新的「思考模式」
- 學習對生成結果做通盤考量
- 每次接龍都是對的，不一定結果是最好的

例如：《天龍八部》珍瓏棋局

InstructGPT: https://arxiv.org/pdf/2203.02155



Transformer in Time-series

Attention is all you need
Vanilla Transformer, NeurIPS 2017

Transformer架构在时间序列预测任务中的初步尝试

Informer, AAAI 2021

Autoformer, NeurIPS 2021

Transformer架构在时间序列预测任务中的改进

FEDformer, ICML 2022

Non-stationary Transformers, NeurIPS 2022

... (Transformer variants)

Transformer究竟对于时间序列预测任务中是否真的有效？ (Are Transformers Effective for Time Series Forecasting?)

Dlinear, AAAI 2023

... (non-Transformer model)

用Series Patching取代data point的为tokenization (PatchTST)

PatchTST, ICLR 2023

... (Patching based Transformer)

LLM for time-series forecasting
Are Language Models Actually Useful for Time Series Forecasting?

图表 7: 预估 LLM 在 GCP TPU v4 芯片上的训练成本

	GPT-3 (OpenAI)	Gopher (Google DeepMind)	MT-NLG (Microsoft/Nvidia)	PaLM (Google Research)
Model Parameters	175B	280B	530B	540B
FLOPs/Token/Model Parameter			6	
TPUs/Machine			4	
Peak FLOPS/TPU			275T	
FLOPS Utilization			46.20%	
Cost/Machine/Hour(1-year reserved)			$8.12	
Seconds/Hour			3600	
Training Cost/1000 Tokens	$0.0047	$0.0075	$0.0141	$0.0144
Train Tokens	300B	300B	270B	780B
Training Cost	$1,398,072	$2,236,915	$3,810,744	$11,216,529

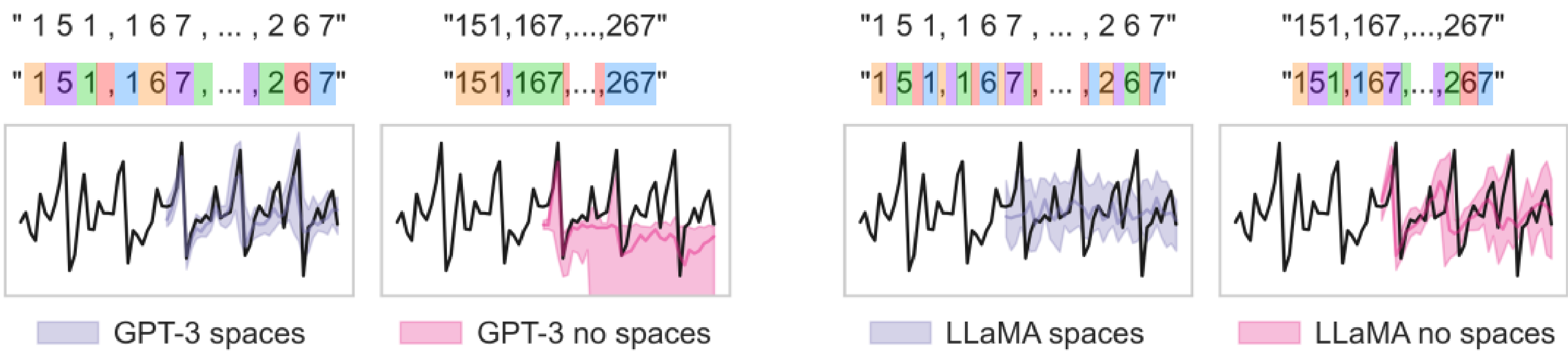
资料来源：OneFlow，国盛证券研究所

Time-series Foundation Model (TSFM) for time-series forecasting

Large Language Models Are Zero-Shot Time Series Forecasters, NeurIPS 2023

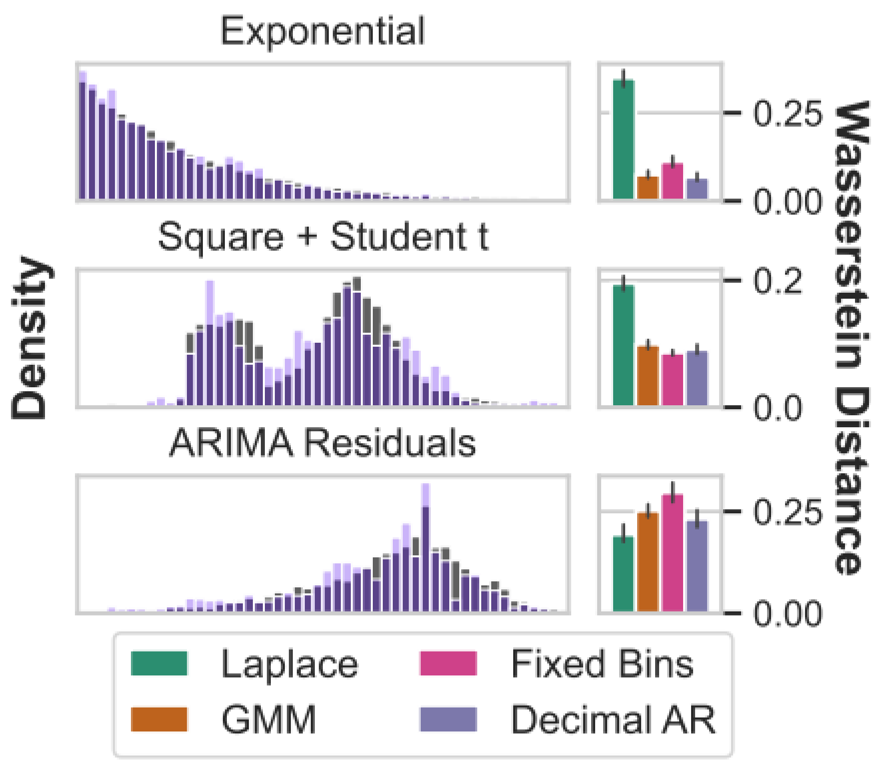
核心思想： 给LLM输入过去序列的数值组成的句子（value to string），不同时间步的数值中间用逗号隔开，让LLM预测未来序列数值。

tokenization：把每个数字看做一个token，对于GPT-3 tokenizer，给数值的每个数之间加上了空格；对于LLaMA，tokenizer默认把每位数当成一个token



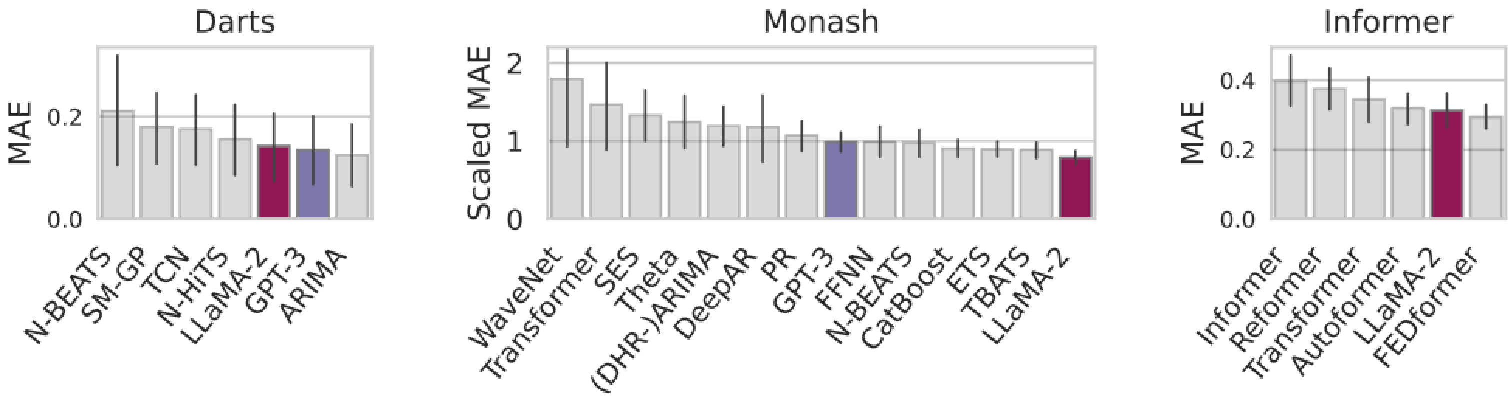
- 因为数值太大会占用很多token造成sequence length过长，所以论文对输入序列rescaling
- 每次预测时都会抽样多次，用预测值中位数或者均值作为点预测结果（这样也可以做概率预测）

为什么LLM能做序列预测？——序列预测本质上就是对预测值的条件分布进行建模（给定已有token条件下预测值的条件概率），作者生成了三种不同分布下，包括指数分布、包含双峰分布的Square+Student t分布以及ARIMA拟合残差分布。LLM整体拟合分布的能力很强，和实际分布的Wasserstein距离较小

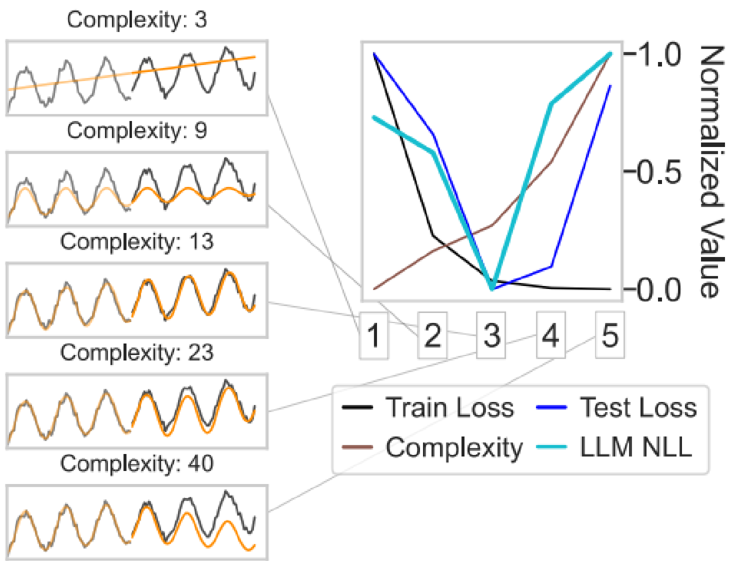


实验结果

整体效果不错，zero-shot情况下就可以超过不少时间序列预测模型

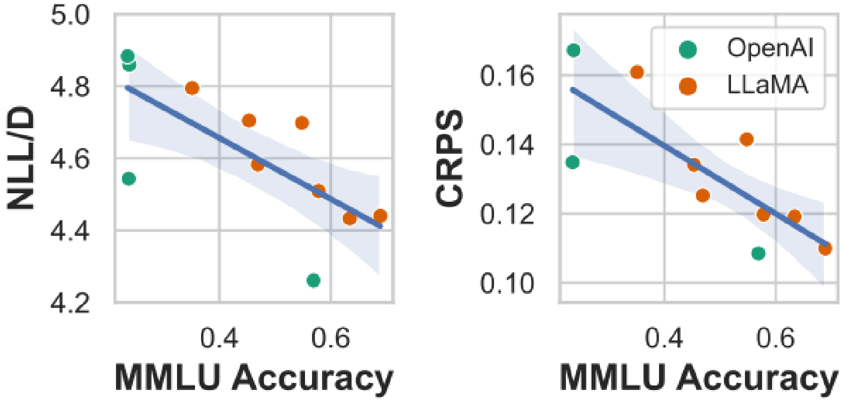


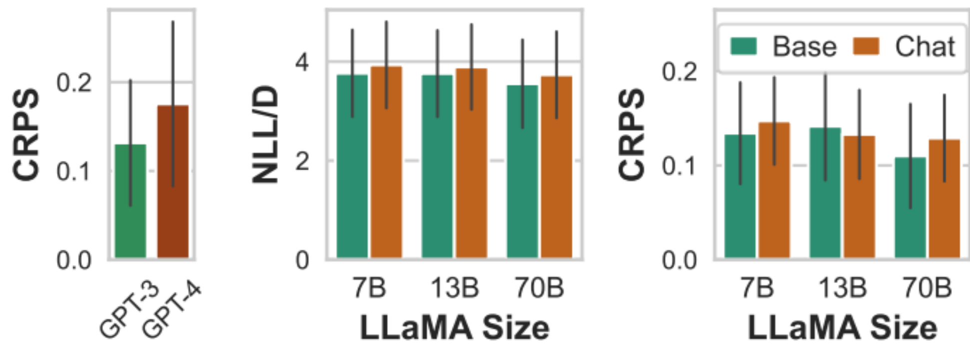
作者用一个case来检验LLM的zero-shot能力，图中是人工生成的序列，可以用不同复杂度的多项式函数来拟合（从欠拟合到过拟合），而LLM去fit这个序列的时候，NLL negative log likelihood最小时对应complexity比较合适的第三种情况。也就是说LLM可以找到数据的低复杂性解释，使得具备zero-shot外推数值序列的能力。



作者把一系列LLM放在一起比较，包括OpenAI models (davinci, babbage, curie, ada), variants of LLaMA (7B, 13B, 33B, 65B) and LLaMA-2 models (7B, 13B, 70B)，结果发现当LLM推理能力更强时，对应预测能力总体也更强，符合直觉

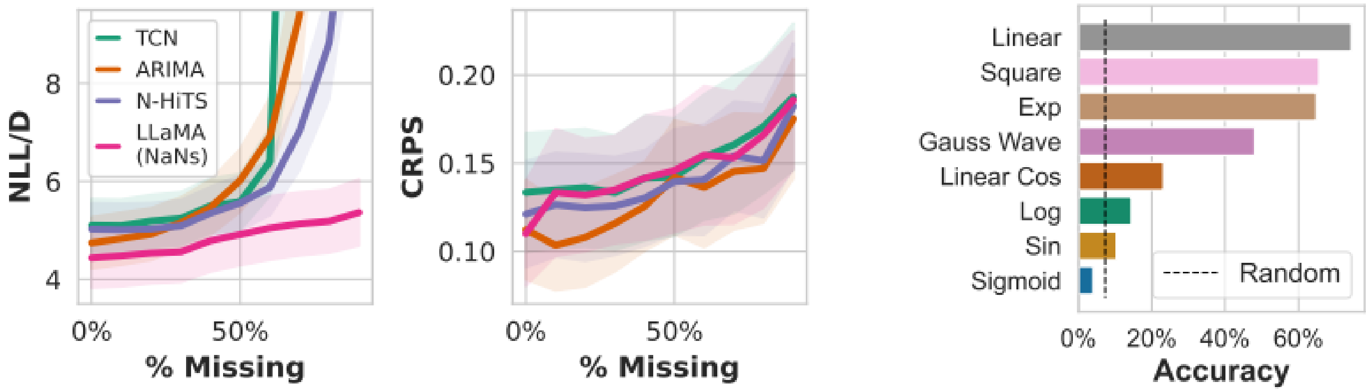
GPT-4的预测能力反而弱于GPT-3，包括LLaMA的Chat版本也是弱于Base版本，说明**RLHF对于LLM本身的序列预测能力可能是有害的**





LLM因为输入是string，可以天然支持NaN，比如输入序列64, NaN, NaN, 49, NaN, 16, NaN，并且相较于linear interpolation+预测模型（TCN、ARIMA以及N-HiTS）而言，不会显著影响模型的NLL

- 另外也支持给定一段生成的时间序列，交互得到哪个生成函数的答案

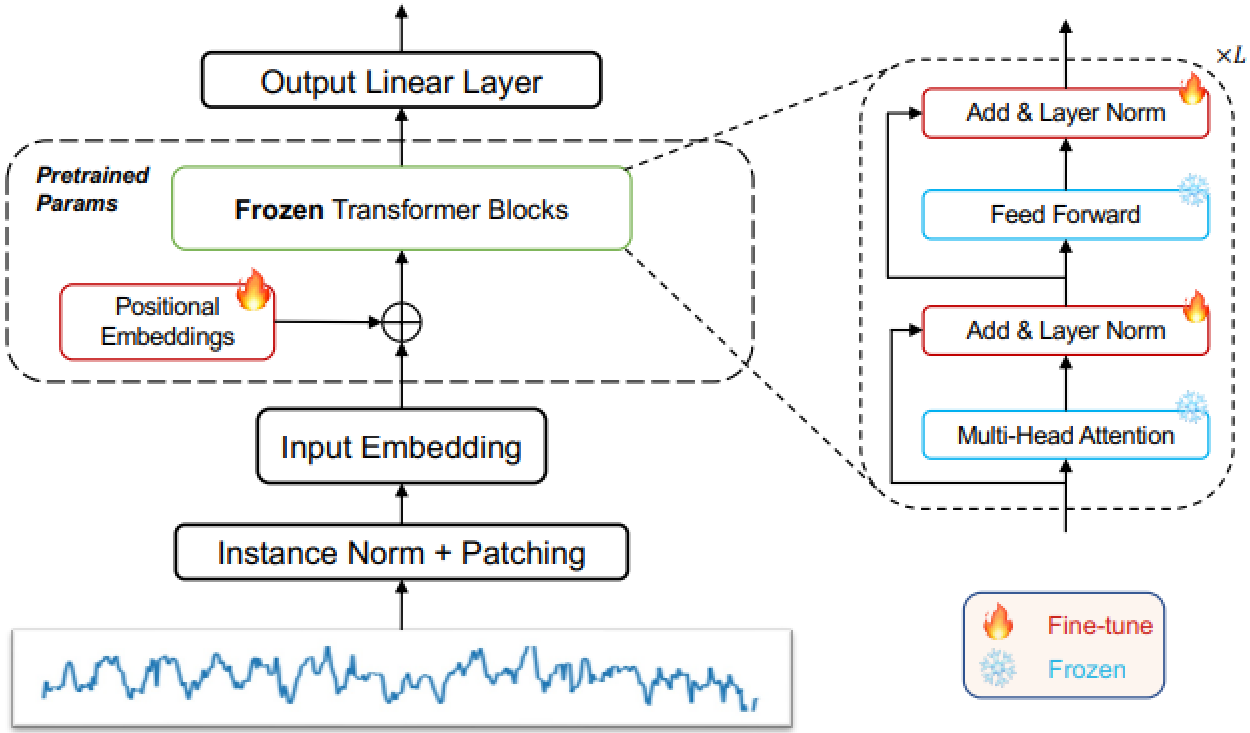


One Fits All: Power General Time Series Analysis by Pretrained LM, NeurIPS 2023

核心思想：LLM的核心在于self-attention和FFN，是通用的知识建模模型，而像embedding层这些都是特定于输入的，输入的域不同自然就要更换，要重新训练。

怎么做？Frozen self-attention以及FFN，微调positional embedding，input embedding，output linear layer以及layer norm

- positional embedding里面有**可学习**的temporal embedding
- NLP任务input embedding是nn.Embedding模块来实现的，对应离散的token index，而现在时间序列输入是**连续的，用nn.linear**
- 区别于上一篇LLMTime，OneFitsAll先将输入序列RevIN做instance norm然后patching看做token

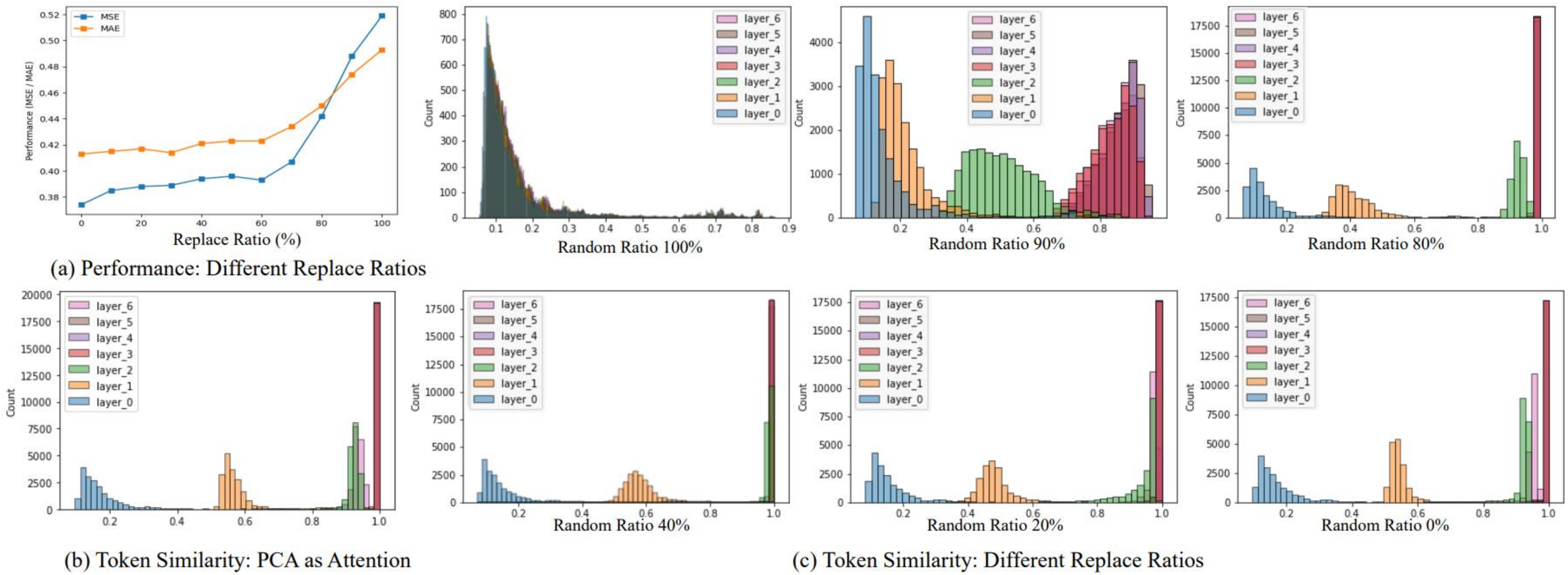


为什么NLP能够迁移在time-series forecasting任务上？

作者将GPT模型中预训练权重按照一定的比例**替换为随机初始化的权重**，子图(a)说明预训练权重被替换越多效果越差（误差越高），符合逻辑

然后作者分析了token之间的相似度，对于模型每一层输出都计算token之间两两相似度

- 全部随机初始化时，token之间相似度非常低
- 完全用预训练权重时（Random Ratio=0%），在较高layer层数上相似度明显非常高，说明token向量都被投影到了输入的低维特征向量空间中。类似于curse of dimension，维度越高计算相似度高的概率指数越低

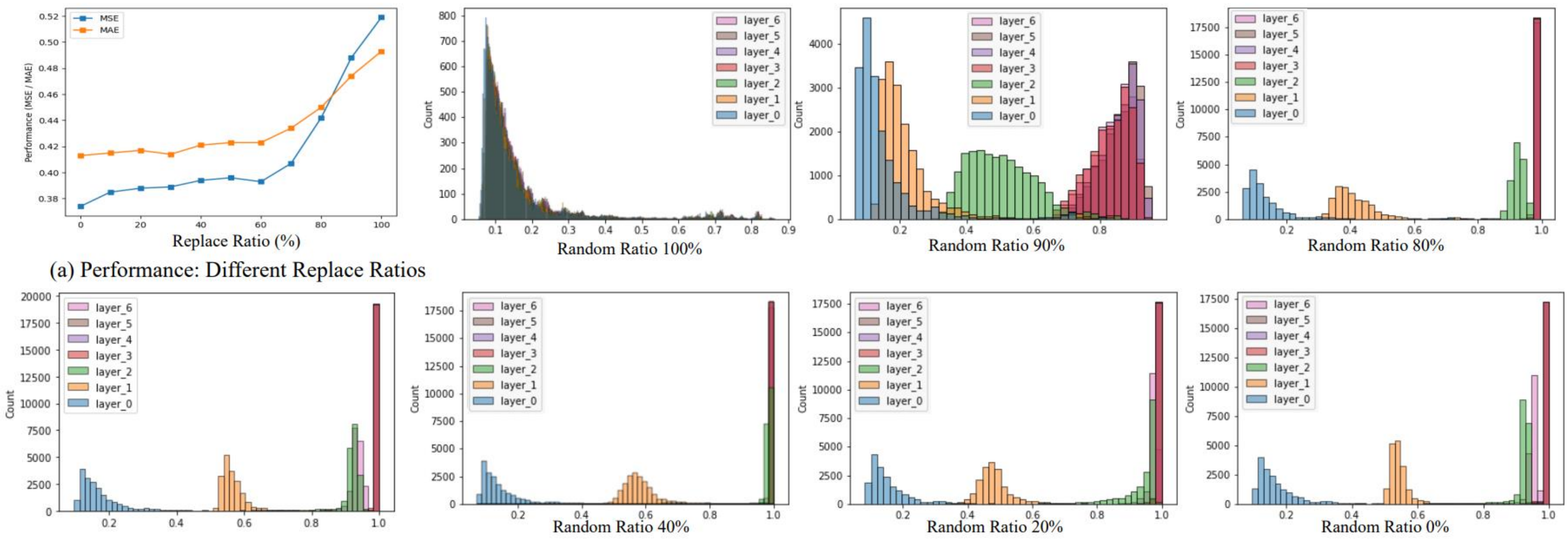


(a) Performance: Different Replace Ratios

(b) Token Similarity: PCA as Attention

(c) Token Similarity: Different Replace Ratios

作者将self-attention替换为PCA，对应子图(b)，结果和Random Ratio=0%，即预训练权重下的self-attention结果非常像，说明两者作用比较相似，也就是类似于PCA，预训练好的self-attention建模各种模态数据具有一定的通用性

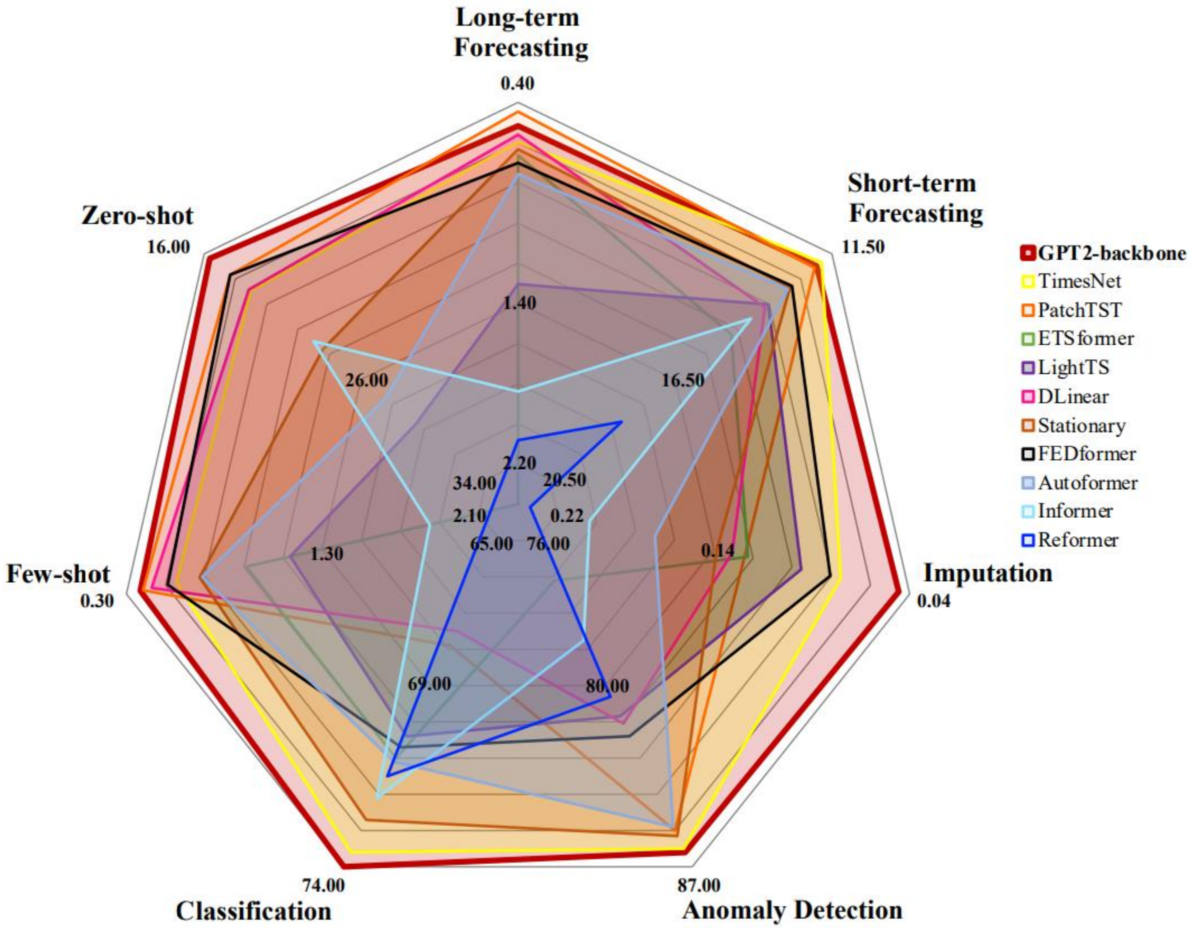


(b) Token Similarity: PCA as Attention

(c) Token Similarity: Different Replace Ratios

实验结果

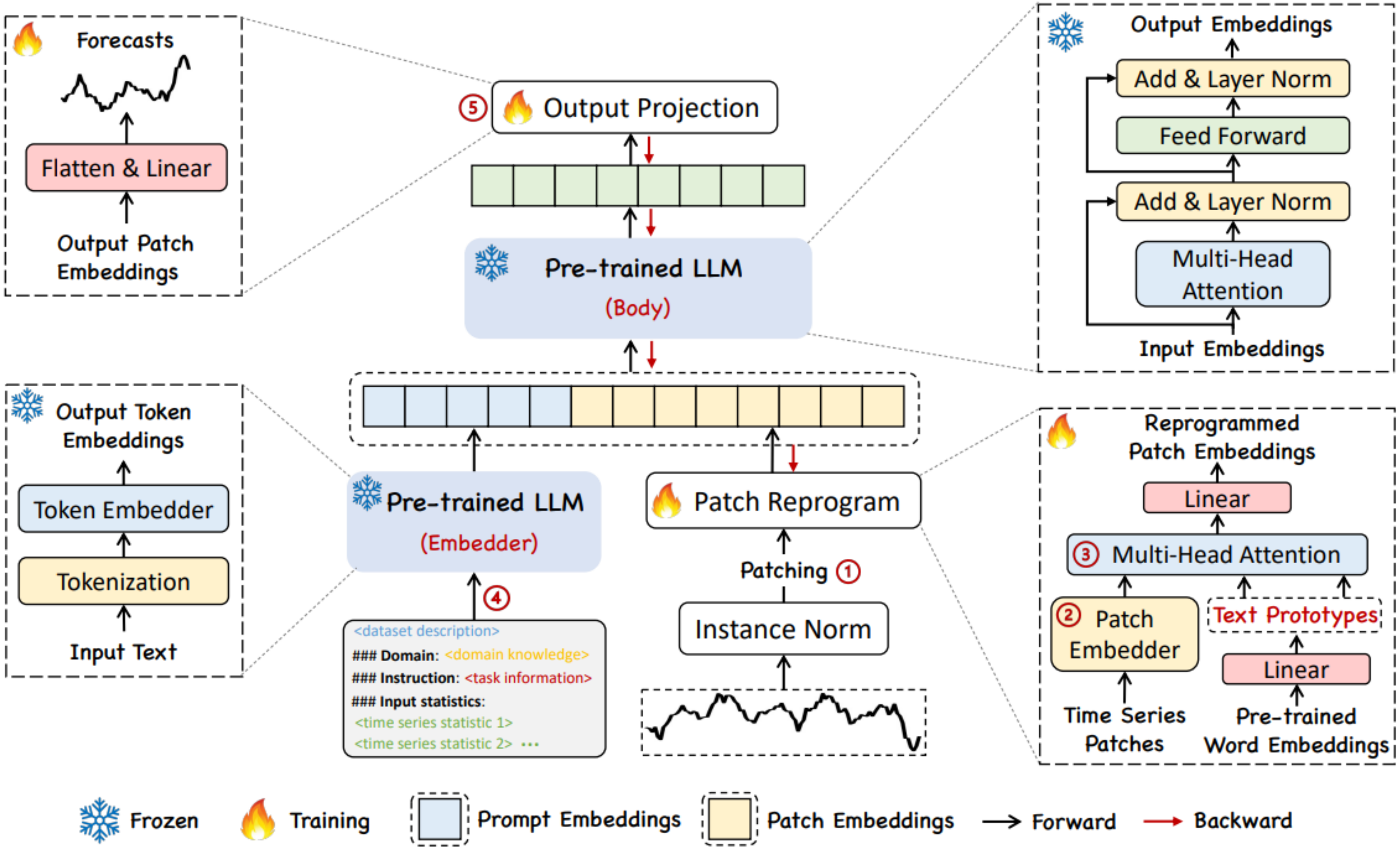
基本各项任务上都不错，并且作者发现用其他的LLM（BERT）甚至是视觉预训练模型（BEiT）或来训练，效果也不错。并且由于大部分层都被冻结，只微调一些参数，因此训练和推理效率也都挺高。



Time-LLM: Time Series Forecasting by Reprogramming Large Language Models, ICLR 2024

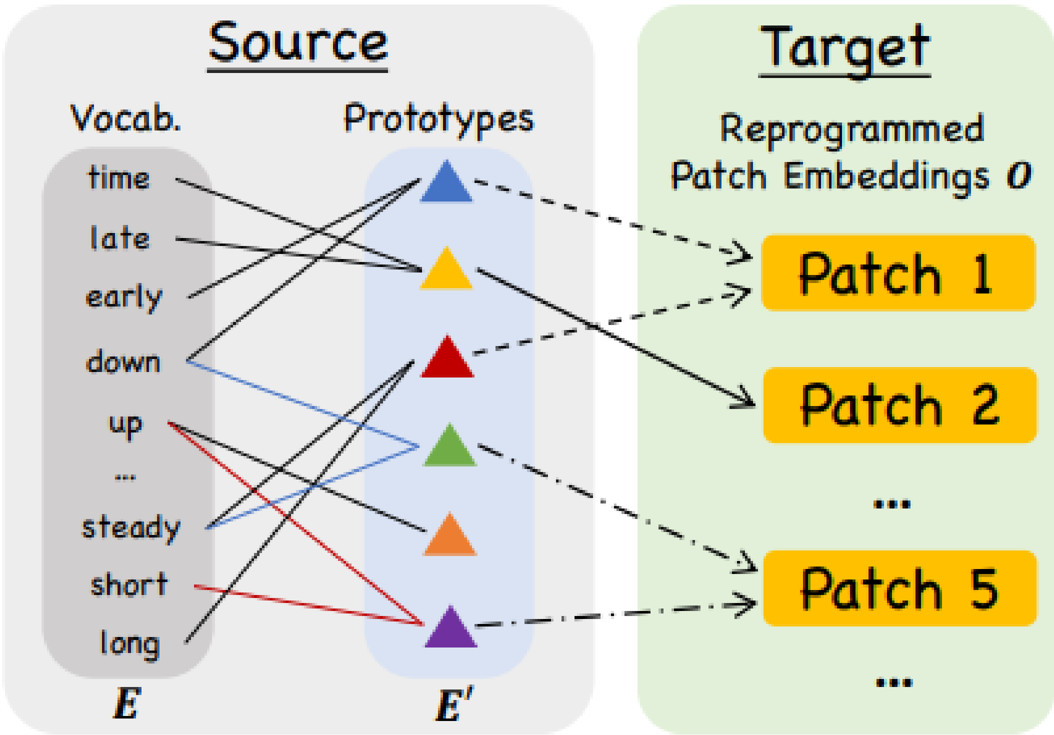
核心思想

•从①开始，time-series先通过RevIN进行instance norm操作，然后patching，得到(BxC)xPxd_m的三维输入，其中B是batch size，C是channel即特征维度（通道独立），P是patch的个数，d_m是定义的隐藏层维度



关键在于patch reprogram，作者通过cross-attention**对齐text2time-series的多模态信息**，以时序embedding作为Query，文本embedding作为Key和Value。但由于NLP词汇表很大，直接做cross-attention对齐**效率不高且稀疏**

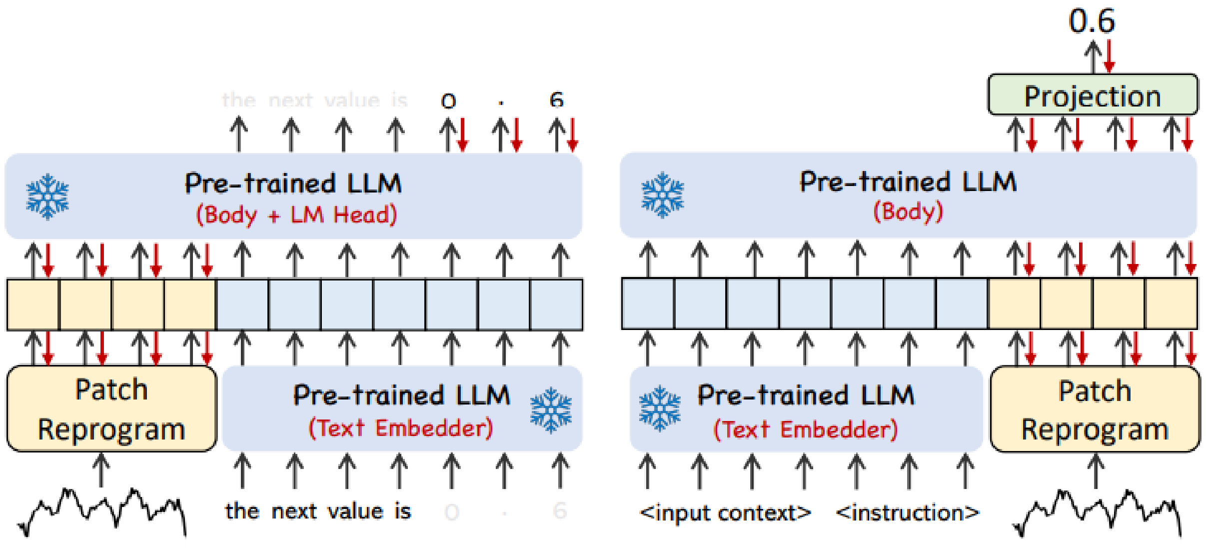
- 作者用了个linear层对text embedding做了projection降维，称之text prototypes，然后用时序embedding和降维后的text embedding做cross-attention
- 举个例子，比如patch 5和绿色、紫色的text prototypes相关性比较大，而紫色的原型主要包含up和short这两个词的信息，绿色的原型主要包含down和steady这两个词的信息，所以第5个patch的时间序列可能包含短期上升趋势和稳定下降趋势



做完Patch Reprogram之后，需要通过LLM得到未来预测值的输出

- Patch-as-Prefix：虽然可以把时序embedding作为前缀prompt，然后和一句文本例如”The next value is ”来得到LLM的预测值，但输出解码起来比较麻烦，而且对于long-term forecasting不是很友好

- Prompt-as-Prefix：paper里面把时间序列数据集的一些统计先验信息作为prompt，和时序embedding拼接喂给LLM，最后接projection layer直接输出预测值



(b) Patch-as-Prefix and Prompt-as-Prefix

The Electricity Transformer Temperature (ETT) indicates the electric power long-term deployment. Each data point consists of the target oil temperature and 6 power load features ...
Below is the information about the input time series:

[BEGIN DATA]
***
[Domain]: We usually observe that electricity consumption peaks at noon, with a significant increase in transformer load
***
[Instruction]: Predict the next <H> steps given the previous <T> steps information attached
***
[Statistics]: The input has a minimum of <min_val>, a maximum of <max_val>, and a median of <median_val>. The overall trend is <upward or downward>. The top five lags are <lag_val>.
[END DATA]

实验结果

- 除了Patch Reprogramming和output projection参数需要训练，其他部分frozen
- long-term forecasting整体效果和PatchTST不相上下
- short-term forecasting效果会更好

Table 1: Long-term forecasting results. All results are averaged from four different forecasting horizons: $H \in \{24, 36, 48, 60\}$ for ILI and $\{96, 192, 336, 720\}$ for the others. A lower value indicates better performance. **Red**: the best, Blue: the second best. Our full results are in Appendix D.

Methods	TIME-LLM (Ours)		GPT4TS (2023a)		DLinear (2023)		PatchTST (2023)		TimesNet (2023)		FEDformer (2022)		Autoformer (2021)		Stationary (2022)		ETSformer (2022)		LightTS (2022a)		Informer (2021)		Reformer (2020)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh1$	**0.408**	**0.423**	0.465	0.455	0.422	0.437	0.413	0.430	0.458	0.450	0.440	0.460	0.496	0.487	0.570	0.537	0.542	0.510	0.491	0.479	1.040	0.795	1.029	0.805
$ETTh2$	0.334	0.383	0.381	0.412	0.431	0.446	**0.330**	**0.379**	0.414	0.427	0.437	0.449	0.450	0.459	0.526	0.516	0.439	0.452	0.602	0.543	4.431	1.729	6.736	2.191
$ETTm1$	**0.329**	**0.372**	0.388	0.403	0.357	0.378	0.351	0.380	0.400	0.406	0.448	0.452	0.588	0.517	0.481	0.456	0.429	0.425	0.435	0.437	0.961	0.734	0.799	0.671
$ETTm2$	**0.251**	**0.313**	0.284	0.339	0.267	0.333	0.255	0.315	0.291	0.333	0.305	0.349	0.327	0.371	0.306	0.347	0.293	0.342	0.409	0.436	1.410	0.810	1.479	0.915
$Weather$	**0.225**	**0.257**	0.237	0.270	0.248	0.300	**0.225**	0.264	0.259	0.287	0.309	0.360	0.338	0.382	0.288	0.314	0.271	0.334	0.261	0.312	0.634	0.548	0.803	0.656
ECL	**0.158**	**0.252**	0.167	0.263	0.166	0.263	0.161	**0.252**	0.192	0.295	0.214	0.327	0.227	0.338	0.193	0.296	0.208	0.323	0.229	0.329	0.311	0.397	0.338	0.422
$Traffic$	**0.388**	0.264	0.414	0.294	0.433	0.295	0.390	**0.263**	0.620	0.336	0.610	0.376	0.628	0.379	0.624	0.340	0.621	0.396	0.622	0.392	0.764	0.416	0.741	0.422
ILI	**1.435**	0.801	1.925	0.903	2.169	1.041	1.443	**0.797**	2.139	0.931	2.847	1.144	3.006	1.161	2.077	0.914	2.497	1.004	7.382	2.003	5.137	1.544	4.724	1.445
1st Count	**7**		0		0		5		0		0		0		0		0		0		0		0	

Table 2: Short-term time series forecasting results on M4. The forecasting horizons are in [6, 48] and the three rows provided are weighted averaged from all datasets under different sampling intervals. A lower value indicates better performance. **Red**: the best, Blue: the second best. More results are in Appendix D.

	Methods	TIME-LLM (Ours)	GPT4TS (2023a)	TimesNet (2023)	PatchTST (2023)	N-HiTS (2023b)	N-BEATS (2020)	ETSformer (2022)	LightTS (2022a)	DLinear (2023)	FEDformer (2022)	Stationary (2022)	Autoformer (2021)	Informer (2021)	Reformer (2020)
Average	SMAPE	**11.983**	12.69	12.88	12.059	12.035	12.25	14.718	13.525	13.639	13.16	12.780	12.909	14.086	18.200
	MASE	**1.595**	1.808	1.836	1.623	1.625	1.698	2.408	2.111	2.095	1.775	1.756	1.771	2.718	4.223
	OWA	**0.859**	0.94	0.955	0.869	0.869	0.896	1.172	1.051	1.051	0.949	0.930	0.939	1.230	1.775

由于Patch Reprogramming和Output Projection参数较少，微调它们可以只用少量的时间序列样本，所以相当于是一个few-shot的解决方案，因而在只用10%训练数据的few-shot场景下TimeLLM表现更好

Table 3: Few-shot learning on 10% training data. We use the same protocol in Tab. 1. All results are averaged from four different forecasting horizons: $H \in \{96, 192, 336, 720\}$. Our full results are in Appendix E.

Methods	TIME-LLM (Ours)		GPT4TS (2023a)		DLinear (2023)		PatchTST (2023)		TimesNet (2023)		FEDformer (2022)		Autoformer (2021)		Stationary (2022)		ETSformer (2022)		LightTS (2022a)		Informer (2021)		Reformer (2020)	
Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh1$	**0.556**	**0.522**	0.590	0.525	0.691	0.600	0.633	0.542	0.869	0.628	0.639	0.561	0.702	0.596	0.915	0.639	1.180	0.834	1.375	0.877	1.199	0.809	1.249	0.833
$ETTh2$	**0.370**	**0.394**	0.397	0.421	0.605	0.538	0.415	0.431	0.479	0.465	0.466	0.475	0.488	0.499	0.462	0.455	0.894	0.713	2.655	1.160	3.872	1.513	3.485	1.486
$ETTm1$	**0.404**	**0.427**	0.464	0.441	0.411	0.429	0.501	0.466	0.677	0.537	0.722	0.605	0.802	0.628	0.797	0.578	0.980	0.714	0.971	0.705	1.192	0.821	1.426	0.856
$ETTm2$	**0.277**	**0.323**	0.293	0.335	0.316	0.368	0.296	0.343	0.320	0.353	0.463	0.488	1.342	0.930	0.332	0.366	0.447	0.487	0.987	0.756	3.370	1.440	3.978	1.587
$Weather$	**0.234**	**0.273**	0.238	0.275	0.241	0.283	0.242	0.279	0.279	0.301	0.284	0.324	0.300	0.342	0.318	0.323	0.318	0.360	0.289	0.322	0.597	0.495	0.546	0.469
ECL	**0.175**	0.270	0.176	**0.269**	0.180	0.280	0.180	0.273	0.323	0.392	0.346	0.427	0.431	0.478	0.444	0.480	0.660	0.617	0.441	0.489	1.195	0.891	0.965	0.768
$Traffic$	**0.429**	0.306	0.440	0.310	0.447	0.313	0.430	**0.305**	0.951	0.535	0.663	0.425	0.749	0.446	1.453	0.815	1.914	0.936	1.248	0.684	1.534	0.811	1.551	0.821
1[st]Count	**7**		1		0		1		0		0		0		0		0		0		0		0	

课程汇报

• 第一组	• Informer
• 第二组	• Dlinear
• 第三组	• PatchTST
• 第四组	• Are Language Models Actually Useful for Time Series Forecasting?

课程汇报需要的part：

- Paper Title, conference, author + affiliation
- Core idea
- 实验结果
- 其他有趣的分析或case study （如果有）
- 你的反思和收获

其他要求

- PPT形式（介绍你们是第几组，姓名，学院专业）
- 控制在30分钟左右

友情提示

- 分组情况
- 可以通过大模型（chatgpt, kimi…）+ 知乎blog来辅助理解