一、引言

1.1 Gemma

2 月 21 日，Google 推出新一代开放 AI 模型——Gemma（https://ai.google.dev/gemma），这是一个轻量级的模型，比肩 Meta 的Llama 2 模型。仅是从名称上来看，最新推出的 Gemma 和此前 Google 的 Gemini 还有点傻傻分不清。对此，Google 也在官宣公告中解释称，Gemma 设计的灵感就是来源于 Gemini，拉丁语 Gemma，有“宝石”之意。二者之间稍有不同的是，可以将 Gemma 视为 Gemini 的更小、更轻的版本。Gemma 的设计目的是让开发人员和研究人员更容易访问和使用，而 Gemini 的设计目的是用于更复杂的任务。两种型号均可免费使用，但 Gemma 的免费套餐更为有限。谷歌开源的Gemma系列模型，在建立了2B和7B两个模版知乎发布在了始智AI wisemodel.cn开源社区。

1.2 Llama-3

4 月 18 日，Meta 发布了 Llama 3，系列大型语言模型（LLM），这是一套经过预训练和指令调优的生成式文本模型，包含8B和70B两个规模。Llama 3 提供两个版本：8B 版本适合在消费级 GPU 上高效部署和开发；70B 版本则专为大规模 AI 应用设计。每个版本都包括基础和指令调优两种形式。Llama 3是一种自回归语言模型，使用优化的Transformer架构。调优版本使用监督微调（SFT）和人类反馈的强化学习（RLHF）来调整模型，以符合人类对实用性和安全性的偏好。

1.3 Phi-3

4月23日，微软研究院公布了Phi-3系列AI模型。这一系列模型包含三个版本：mini(3.8B参数)、small(7B参数)以及medium(14B参数)，分别满足不同场景和需求的应用。其技术报告的副标题是：A Highly Capable Language Model Locally on Your Phone。Phi-3-mini也是小型模型中的第一款，其训练数据集比GPT-4 等大型语言模型要小，可在 Azure、Hugging Face 和 Ollama 上使用。尽管phi-3-mini的规模小到可以轻松地在现代手机上运

行，但它的性能与Mixtral 8x7B和GPT-3.5等模型相当。

二、评测数据

2.1 Gemma

Gemma模型在多个基准测试中展现了出色的性能，不仅在其规模上实现了同类最佳的性能，甚至在某些任务上超越了规模更大的模型。这一性能的提升，使Gemma在数学、Python代码生成、常识和常识推理等任务上具有显著的应用潜力。谷歌声称，Gemma 模型18个语言理解、推理、数学等关键基准测试中，有11个测试分数超越了Meta Llama-2等更大参数的开源模型。平均分数方面，Gemma -7B的基准测试平均分高达56.4，远超过Llama-13B（52.2）、Mistral-7B（54.0），成为目前全球最强大的开源模型。Gemma（7B）与 LLaMA 2（7B）、LLaMA 2（13B）和 Mistral（7B）在问答、推理、数学和科学、编码等任务上的性能比较。可以看到，Gemma（7B）表现出了优势（除了在问答任务上弱于 LLaMA 2（13B））。

2.2 Llama-3

Llama 3 的基准测试表明，tokenizer 提高了 token 化效率，与 Llama 2 相比，token 生成量最多可减少 15%。此外，组查询关注（GQA）现在也被添加到了 Llama 3 8B。扎克伯格表示，Llama 3的训练效率比 Llama 2 高 3 倍，基于超过 15T token 训练，相当于 Llama 2 数据集的 7 倍还多，同时带有 Llama Guard 2、Code Shield 和 CyberSec Eval 2 的新版信任和安全工具。最后支持 8K 长文本，改进的 tokenizer 具有 128K token 的词汇量，可实现更好的性能。

2.3 Phi-3

Phi-3的话

三、模型效果

3.1 Gemma

Google的 Gemma 2B 和 7B适合简单的聊天机器人和语言相关工作，非常适用于各种文本生成任务，包括问答、总结和推理。它们相对较小的体积使其可以部署在资源有限的环境中，如笔记本电脑、台式机或您自己的云基础设施，实现了对最先进AI模型的民主化访问。

另外，Gemma 7B 其实是 8.5B。开发者使用监督式微调（SFT）对 Gemma 2B 和 7B 进行微调，微调内容包括纯文本、仅英语的合成以及人工生成的提示-响应对。同时，我们还利用基于人类反馈的强化学习（RLHF），在此过程中，奖励模型训练基于标记的仅英语偏好数据，而策略则基于一组高质量的提示。我们发现，这两个阶段对于提升模型在下游自动评估和人类偏好评估中的表现都非常重要。本地运行Gemma 2B和7B，一般对Gemma 7B的量化，再使用QLORA对Gemma 7B进行微调。

Gemma 模型在 6 个标准安全基准和人类并列评估中优于竞争对手，在对话、推理、数学和代码生成等广泛领域的性能得到提升。MMLU (64.3%) 和 MBPP (44.4%) 的结果证明了 Gemma 的高性能，以及开放式 LLM 性能的持续发展空间。

最后可以将模型部署到如推理阶段，进行提示过滤等工作，涉及到像Llama Guard和Code Shield类似的工具。

	Phi-3-mini 3.8b	Phi-3-small 7b	Phi-3-medium 14b (preview)	Phi-2 2.7b	Mistral 7b	Gemma 7b	Llama-3-In 8b	Mixtral 8x7b	GPT-3.5 version 1106
MMLU (5-Shot) [HBK+21]	68.8	75.3	78.2	56.3	61.7	63.6	66.0	68.4	71.4
HellaSwag (5-Shot) [ZHB+19]	76.7	78.7	83.0	53.6	58.5	49.8	69.5	70.4	78.8
ANLI (7-Shot) [NWD+20]	52.8	55.0	58.7	42.5	47.1	48.7	54.8	55.2	58.1
GSM-8K (0-Shot; CoT) [CKB+21]	82.5	88.9	90.3	61.1	46.4	59.8	77.4	64.7	78.1
MedQA (2-Shot) [JPO+20]	53.8	58.2	69.4	40.9	49.6	50.0	58.9	62.2	63.4
AGIEval (0-Shot) [ZCG+23]	37.5	45.0	48.4	29.8	35.1	42.1	42.0	45.2	48.4
TriviaQA (5-Shot) [JCWZ17]	64.0	59.1	75.6	45.2	72.3	75.2	73.6	82.2	85.8
Arc-C (10-Shot) [CCE+18]	84.9	90.7	91.0	75.9	78.6	78.3	80.5	87.3	87.4
Arc-E (10-Shot) [CCE+18]	94.6	97.1	97.8	88.5	90.6	91.4	92.3	95.6	96.3
PIQA (5-Shot) [BZGC19]	84.2	87.8	87.7	60.2	77.7	78.1	77.1	86.0	86.6
SociQA (5-Shot) [BZGC19]	76.6	79.0	80.2	68.3	74.6	65.5	73.2	75.9	68.3
BigBench-Hard (0-Shot) [SRR+22, SSS+22]	71.7	75.0	81.3	59.4	57.3	59.6	68.9	69.7	68.32
WinoGrande (5-Shot) [SLBBC19]	70.8	82.5	81.4	54.7	54.2	55.6	58.0	62.0	68.8
OpenBookQA (10-Shot) [MCKS18]	83.2	88.4	87.2	73.6	79.8	78.6	81.6	85.8	86.0
BoolQ (0-Shot) [CLC+19]	77.2	82.9	86.6	–	72.2	66.0	78.3	77.6	79.1
CommonSenseQA (10-Shot) [THLB19]	80.2	80.3	82.6	69.3	72.6	76.2	73.6	78.1	79.6
TruthfulQA (10-Shot) [LHE22]	65.0	68.7	75.7	–	52.1	53.0	62.0	60.1	85.8
HumanEval (0-Shot) [CTJ+21]	58.5	59.1	55.5	59.0	28.0	34.1	38.4	37.8	62.2
MBPP (3-Shot) [AON+21]	70.0	71.4	74.5	60.6	50.8	51.5	65.3	60.2	77.8
Average	71.2	74.9	78.2	–	61.0	62.0	68.0	69.9	75.3
GPQA (2-Shot; CoT) [RHS+23]	32.8	34.3	–	–	–	–	–	–	29.0
MT Bench (2 round ave.) [ZCS+23]	8.38	8.70	8.91	–	–	–	–	–	8.35

3.2 Llama-3

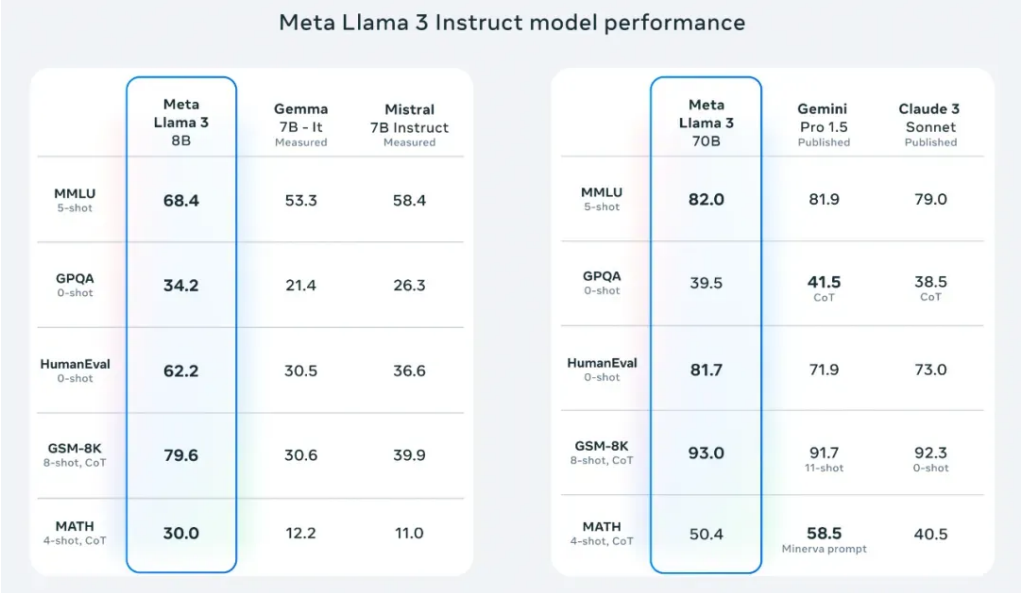
Meta 最近发布的 Llama 3 8B可能会用于一些聊天机器人和编码辅助工作。由于训练过程使用了超过15万亿的token，因此需要大量的计算资源，团队自己搭建了计算集群（两个24k H100 GPU）用于训练模型。Joe强调，目前发布的其实是Llama 3的非常早期版本，团队原本打算将这些模型称为预发布或预览版本，因为模型并不具有计划中包含的全部功能。Llama 3 还包括了更大的词汇表，一个新的tokenizer，运行效率更高，性能更强，并且上下文窗口也加倍了。Meta在未来的研究方向是开发出紫色的Llama（融合了红色和蓝色），即红队和蓝队，也就是攻击方和防御方，开发团队从网络安全领域借鉴了命名方式，也是内部网络安全/生成式AI团队的一位科学家提出的。

为了训练最好的语言模型，Meta团队扩大了人工标注SFT数据的规模（1000万），将GPU数量也扩大到了数万个，还采用了诸如拒绝采样、PPO、DPO等技术来尝试在这些模型的可用性、人类特征以及预训练中的大规模数据之间找到平衡。为了确保 Llama 3 接受最高质量数据的训练，研究团队开发了一系列数据过滤 pipeline，包括使用启发式过滤器

（filter）、NSFW 过滤器、语义重复数据删除方法和文本分类器来预测数据质量。torchtune是一个纯粹的PyTorch微调库，可以很容易地对LLM进行微调，没有各种依赖项，支持Llama 3，目前已经与HuggingFace和其他一些库进行了集成。

基准测试表明，tokenizer 提高了 token 化效率，与 Llama 2 相比，token 生成量最多可减少 15%。此外，组查询关注（GQA）现在也被添加到了 Llama 3 8B。Llama 3 使用从公开来源收集的超过 15T 的 token 进行了预训练，是 Llama 2 使用的数据集的七倍多，并且包含四倍多的代码，其中超过 5% 的 Llama 3 预训练数据集由涵盖 30 多种语言的高质量非英语数据组成。因此，尽管与 Llama 2 7B 相比，该模型多了 1B 个参数，但 tokenizer 效率和 GQA 的改进有助于保持与 Llama 2 7B 相当的推理效率。研究团队还进行了广泛的实验，以评估出在最终预训练数据集中不同来源数据的最佳混合方式，最终确保 Llama 3 在各种用例（包括日常问题、STEM、编码、历史知识等）中表现良好。研发团队针对后训练模型（即指令模型），以及基础模型本身都进行了评估，可以看到8B和70B的指令模型都优于同级对比模型，基础模型Llama 3 70B在各方面也都优于

Gemini Pro 1.0模型，甚至也优于最近发布的Mistral 8*22B，总之模型的性能表现非常强劲。



3.3 Phi-3

针对发行的三个版本，微软团队修改并利用了有关助益性和无害性的偏好数据集[BJN+22, JLD+23]，以及多个公司内部生成的数据集，以应对安全后期训练中的潜在风险类别。微软内部的独立红队在模型后期训练过程中反复审查phi-3-mini，以识别和改进潜在的风险点。

在内部RAI基准测试中，实验使用GPT-4模拟了五个不同类别的多轮对话，并对模型的回应进行了评估。回应的“依据性”（Ungroundedness）评分介于0（完全有依据）到4（完全无依据）之间，以判断回应信息是否基于给定的提示。在其他类别中，根据回应可能导致的危害严重性，评分从0（无害）到7（极度危害）。缺陷率（DR-x）则是将严重性评分达到或超过x的样本百分比计算出来。小模型能力已经超过了chatgpt，未来几个月内会超过gpt4，如果能力相同，小模型的成本优势巨大。phi-3模型现在在知识记忆上比较弱，但是可以通过rag来解决，还可以降低大模型的幻觉。大模型和小模型定位不一样的，大模型参数多，通用AI，小模型参数少，基于企业自有知识的建模。

Category	Benchmark	Phi-3				Gemma-7b	Mistral-7b	Mixtral-8x7b	Llama-3-8B-In	GPT3.5-Turbo-1106	Claude-3 Sonnet
		Phi-3-Mini-4K-In	Phi-3-Mini-128K-In	Phi-3-Small (Preview)	Phi-3-Medium (Preview)						
Popular Aggregate Benchmarks	AGI Eval (0-shot)	37.5	36.9	45	48.4	42.1	35.1	45.2	42	48.4	48.4
	MMLU (5-shot)	68.8	68.1	75.6	78.2	63.6	61.7	70.5	66.5	71.4	73.9
	BigBench Hard (0-shot)	71.7	71.5	74.9	81.3	59.6	57.3	69.7	51.5	68.3	--
Language Understanding	ANLI (7-shot)	52.8	52.8	55	58.7	48.7	47.1	55.2	57.3	58.1	68.6
	HellaSwag (5-shot)	76.7	74.5	78.7	83	49.8	58.5	70.4	71.1	78.8	79.2
Reasoning	ARC Challenge (10-shot)	84.9	84	90.7	91	78.3	78.6	87.3	82.8	87.4	91.6
	ARC Easy (10-shot)	94.6	95.2	97.1	97.8	91.4	90.6	95.6	93.4	96.3	97.7
	BoolQ (0-shot)	77.6	78.7	82.9	86.6	66	72.2	76.6	80.9	79.1	87.1
	CommonsenseQA (10-shot)	80.2	78	80.3	82.6	76.2	72.6	78.1	79	79.6	82.6
	MedQA (2-shot)	53.8	55.3	58.2	69.4	49.6	50	62.2	60.5	63.4	67.9
	OpenBookQA (10-shot)	83.2	80.6	88.4	87.2	78.6	79.8	85.8	82.6	86	90.8
	PIQA (5-shot)	84.2	83.6	87.8	87.7	78.1	77.7	86	75.7	86.6	87.8
	Social IQA (5-shot)	76.6	76.1	79	80.2	65.5	74.6	75.9	73.9	68.3	80.2
	TruthfulQA (MC2) (10-shot)	65	63.2	68.7	75.7	52.1	53	60.1	63.2	67.7	77.8
	WinoGrande (5-shot)	70.8	72.5	82.5	81.4	55.6	54.2	62	65	68.8	81.4
Factual Knowledge	TriviaQA (5-shot)	64	57.1	59.1	75.6	72.3	75.2	82.2	67.7	85.8	65.7
Math	GSM8K Chain of Thought (0-shot)	82.5	83.6	88.9	90.3	59.8	46.4	64.7	77.4	78.1	79.1
Code generation	HumanEval (0-shot)	59.1	57.9	59.1	55.5	34.1	28	37.8	60.4	62.2	65.9
	MBPP (3-shot)	53.8	62.5	71.4	74.5	51.5	50.8	60.2	67.7	77.8	79.4

四、模型解析与部署

4.1 Gemma

4.1.1 模型介绍

Gemma系列模型是在 Google DeepMind 和其他 Google AI 团队共同开发而成，采用与 Gemini 模型相同的研究和技术，建立在序列模型、Transformer、基于神经网络的深度学习方法和分布式系统上大规模训练技术至上。模型训练的上下文长

度为 8192 个 token。这些是文本到文本的、仅解码器的大型语言模型，提供英语版本，开放权重，预训练变体和指令调优变体。

模型内部，Gemma使用最新一代的张量处理单元硬件（TPUv5e）进行训练。 训练大型语言模型需要大量的计算能力。TPUs专为机器学习中常见的矩阵操作设计，提供了几个方面的优势：为处理训练LLMs中涉及的大量计算，与CPU相比，TPU它们可以大幅加速训练，通常配备大量高带宽内存，允许在训练过程中处理大型模型和批量大小。TPU Pods（大型TPU集群）为处理大型基础模型的不断增长的复杂性提供了可扩展的解决方案，可以在多个TPU设备上分布训练，以实现更快、更高效的处理。在许多场景中，与基于CPU的基础设施相比，TPUs是考虑到由于训练速度更快而节省的时间和资源。

Gemma 与其他架构之间的一大区别是它使用了 GeGLU 激活，而 GeGLU 激活是在 2020 年的谷歌论文《GLU Variants Improve Transformer》中提出的。此外，目前 7B 大小规模的开源模型已经有很多了，因此 Gemma 2B 更加有趣，它可以轻松地在单个 GPU 上运行。

此外，训练使用了JAX和ML Pathways。JAX允许研究人员利用最新一代硬件，包括TPUs，ML Pathways是能够跨多个任务泛化人工智能系统。两个工具一起使用，正如在关于Gemini模型系列的论文中所描述的；"Jax和Pathways的‘单一控制器’编程模型允许一个Python进程来指挥整个训练运行，极大地简化了开发工作流程。"通常使用Hugging Face TRL + Colab GPU，或者Keras（JAX 后端） + Kaggle TPU -> Hugging Face的组合，搭配使用苹果 MLX 框架较多。

4.1.2 模型部署

运用ollama:
ollama run gemma:2b
ollama run gemma:7b (default)

运用Google Colaboratory:
!pip install -U keras-nlp
!pip install -q -U keras
!pip install protobuf --quiet
!pip install apache_beam[gcp] --quiet

```
!pip install keras_nlp==0.8.0 --quiet

# To use the newly installed versions, restart the runtime.
exit()

import keras
import keras_nlp
import numpy as np

生成文本
gemma_lm=keras_nlp.models.GemmaCausalLM.from_preset("gemma_instruct_2b_en")
gemma_lm.generate("Keras is a", max_length=30)
gemma_lm.generate(["Keras is a", "I want to say"], max_length=30)

自定义采样器生成文本
gemma_lm=keras_nlp.models.GemmaCausalLM.from_preset("gemma_instruct_2b_en")
gemma_lm.compile(sampler="top_k")
gemma_lm.generate("I want to say",
```

max_length=30)
gemma_lm.compile(sampler=keras_nlp.samplers.BeamSampler(num_beams=2))
gemma_lm.generate("I want to say", max_length=30)

不预处理直接生成
prompt = {"token_ids": np.array([[2, 214064, 603, 0, 0, 0, 0]] * 2),"padding_mask": np.array([[1, 1, 1, 0, 0, 0, 0]] * 2)}
gemma_lm=keras_nlp.models.GemmaCausalLM.from_preset("gemma_instruct_2b_en",
 preprocessor=None)
gemma_lm.generate(prompt)

用单支数据去拟合
features = ["The quick brown fox jumped.", "I forgot my homework."]
gemma_lm=keras_nlp.models.GemmaCausalLM.from_preset("gemma_instruct_2b_en")
gemma_lm.fit(x=features, batch_size=2)

不预处理从单支数据去拟合

```
x={"token_ids": np.array([[2, 214064, 603, 5271, 6044, 9581, 3, 0]] * 2),"padding_mask": np.array([[1, 1, 1, 1, 1, 1, 1, 0]] * 2), }
y = np.array([[214064, 603, 5271, 6044, 9581, 3, 0, 0]] * 2)
sw = np.array([[1, 1, 1, 1, 1, 1, 0, 0]] * 2)
gemma_lm=keras_nlp.models.GemmaCausalLM.from_preset("gemma_instruct_2b_en",preprocessor=None)
gemma_lm.fit(x=x,y=y,sample_weight=sw,batch_size=2)
```

自定义个推理函数

```
def gemma_inference_function(model, batch, inference_args, model_id):
  vectorized_batch = np.stack(batch, axis=0)
   # The only inference_arg expected here is a max_length parameter to
  # determine how many words are included in the output.
                                predictions = model.generate(vectorized_batch, **inference_args)
```

```
        return utils._convert_to_result(batch,
predictions, model_id)

执行流水线
class FormatOutput(beam.DoFn):
  def process(self, element, *args, **kwargs):
                yield "Input: {input}, Output:
{output}".format(input=element.example,
output=element.inference)

# Instantiate a NumPy array of string prompts
for the model.
examples = np.array(["Tell me the sentiment of
the phrase 'I like pizza': "])
# Specify the model handler, providing a path
and the custom inference function.
model_handler                                  =
TFModelHandlerNumpy(model_path,
inference_fn=gemma_inference_function)
with beam.Pipeline() as p:
   _ = (p | beam.Create(examples) # Create a
PCollection of the prompts.
                  | RunInference(model_handler,
```

```
inference_args={'max_length': 32}) # Send the prompts to the model and get responses.
        | beam.ParDo(FormatOutput()) # Format the output.
        | beam.Map(print) # Print the formatted output.
    )
```

4.2 Llama-3

4.2.1 模型介绍

Llama 3使用的是稠密自回归Transformer，在模型中加入了群组查询注意力（grouped query attention，GQA）机制，又添加了一个新的分词器，团队表示会在即将发布的论文中详细介绍这个问题。他的指令调优模型针对对话用例进行了优化，在许多常见的行业基准测试中，其性能优于许多现有的开源聊天模型。此外，在开发这些模型的过程中，我们非常重视优化模型的实用性和安全性。微调数据包括公开可用的指令数据集，以及超过1000万个人工标注的示例。

4.2.2 模型部署

gguf格式推理：

```
curl -fsSL https://ollama.com/install.sh | sh
ollama run mistral
sudo apt update && sudo apt install git g++ wget build-essential
git clone https://github.com/ggerganov/llama.cpp
cd llama.cpp
cd models
wgethttps://huggingface.co/TheBloke/Mistral-7B-v0.1-GGUF/resolve/main/mistral-7b-v0.1.Q4_K_S.gguf
cd ..
./main -m models/mistral-7b-v0.1.Q4_K_S.gguf -p "Whatsup?" -n 400 -e
```

ollama结合本地推理：

```
!pip install ollama
import ollama
response = ollama.chat(model='llama3',messages=
```

```
[{'role': 'user','content': 'Why is the sky blue?',},]
)
print(response['message']['content'])
stream = ollama.chat(model='llama3',messages
=
[{'role': 'user', 'content': 'Why is the sky blue?'}],
stream=True,)
for chunk in stream:
print(chunk['message']
['content'], end='', flush=True)
ollama.chat(model='llama3', messages=
[{'role': 'user', 'content': 'Why is the sky blue?'}])
ollama.generate(model='llama3', prompt='Why
is the sky blue?')
ollama.list()
modelfile='''
FROM llama3
SYSTEM You are mario from super mario bros.
'''
ollama.create(model='example', modelfile=mo
delfile)
ollama.copy('llama3', 'user/llama3')
ollama.delete('llama3')
```

```
Pull
ollama.pull('llama3')
push
ollama.push('user/llama3')
ollama.embeddings(model='llama3', prompt='The sky is blue because of rayleigh scattering')
from ollama import Client
client = Client(host='http://localhost:11434')
response = client.chat(model='llama3', messages=
[{'role': 'user','content': 'Why is the sky blue?',},]
)
import asyncio
from ollama import AsyncClient
async def chat():
message = {'role': 'user', 'content': 'Why is the sky blue?'}
response = await AsyncClient().chat(model='llama3', messages=[message])
asyncio.run(chat())
import asyncio
from ollama import AsyncClient
async def chat():
```

```
message = {'role': 'user', 'content': 'Why is the sky blue?'}
async for part in await AsyncClient().chat(model='llama3', messages=[message], stream=True):
print(part['message']
['content'], end='', flush=True)
asyncio.run(chat())
model='does-not-yet-exist'
try:
ollama.chat(model)
except ollama.ResponseError as e:
print('Error:', e.error)
if e.status_code == 404:
ollama.pull(model)
```

4.3 Phi-3

4.3.1 模型介绍

微软Azure人工智能平台公司副总裁埃里克-博伊德（Eric Boyd）介绍说，Phi-3 Mini的性能与GPT-3.5等LLM不相上下，"只是外形尺寸更小而已"。与体积较大的同类，小型人工智能模型的运行成本通常更

低，在手机和笔记本电脑等个人。该公司于 12 月发布了 Phi-2，其性能与 Llama 2 等更大的型号不相上下。微软表示，Phi-3 的性能比前一版本更好，其响应速度接近比它大 10 倍的机型。

据 The Information今年早些时候报道，微软正在组建一个专门研究轻量级人工智能模型的团队。 除了 Phi，该公司还建立了一个专注于解决数学问题的模型Orca-Math。微软的竞争对手也有自己的小型人工智能模型，它们大多针对文档摘要或编码辅助等较简单的任务。Google的 Gemma 2B 和 7B适合简单的聊天机器人和语言相关工作。

微软Azure人工智能平台公司副总裁埃里克-博伊德（Eric Boyd）介绍说，"只是外形尺寸更小而已，在phi-2使用的更大、更先进的数据集上进行了3.3T tokens的训练"。开发人员用"课程"来训练 Phi-3。他们的灵感来自于儿童是如何从睡前故事、单词较简单的书籍以及谈论较大主题的句子结构中学习的。Boyd 还说，公司经常发现，像 Phi-3 这样的小型机型更适合他们的定制应用，因为对于很多公司来说，他们的内部数据集无论如何都会偏小。而且，由于这

些型号使用的计算能力较低，它们的价格往往要低得多。

4.3.2 模型部署

curl -fsSL https://ollama.com/install.sh | sh

推理：

```
import torch
from transformers import AutoModelForCausalLM, AutoTokenizer, pipeline

torch.random.manual_seed(0)

model = AutoModelForCausalLM.from_pretrained(
    "microsoft/Phi-3-mini-4k-instruct",
    device_map="cuda",
    torch_dtype="auto",
    trust_remote_code=True,
)
```

```
tokenizer = AutoTokenizer.from_pretrained("microsoft/Phi-3-mini-4k-instruct")

messages = [
    {"role": "user", "content": "Can you provide ways to eat combinations of bananas and dragonfruits?"},
     {"role": "assistant", "content": "Sure! Here are some ways to eat bananas and dragonfruits together: 1. Banana and dragonfruit smoothie: Blend bananas and dragonfruits together with some milk and honey. 2. Banana and dragonfruit salad: Mix sliced bananas and dragonfruits together with some lemon juice and honey."},
       {"role": "user", "content": "What about solving an 2x + 3 = 7 equation?"},
]

pipe = pipeline(
   "text-generation",
   model=model,
```

```
    tokenizer=tokenizer,
)

generation_args = {
    "max_new_tokens": 500,
    "return_full_text": False,
    "temperature": 0.0,
    "do_sample": False,
}

output = pipe(messages, **generation_args)
print(output[0]['generated_text'])
```

ONNX格式模型推理：

```
git clone https://www.modelscope.cn/LLM-Research/Phi-3-mini-4k-instruct-onnx.git

pip install --pre onnxruntime-genaicurl https://raw.githubusercontent.com/microsoft/onnxruntime-genai/main/examples/python/model-qa.py -o model-qa.py
```

python model-qa.py -m Phi-3-mini-4k-instruct-onnx/cpu_and_mobile/cpu-int4-rtn-block-32 -l 2048

微调1：

<|user|>How to explain Internet for a medieval knight?<|end|>
<|assistant|>

<|user|>
I am going to Paris, what should I see?<|end|>
<|assistant|>
Paris, the capital of France, is known for its stunning architecture, art museums, historical landmarks, and romantic atmosphere. Here are some of the top attractions to see in Paris:\n\n1. The Eiffel Tower: The iconic Eiffel Tower is one of the most recognizable landmarks in the world and offers breathtaking views of the city.\n2. The Louvre Museum: The Louvre is one of the world's largest and most famous museums, housing an impressive

collection of art and artifacts, including the Mona Lisa.\n3. Notre-Dame Cathedral: This beautiful cathedral is one of the most famous landmarks in Paris and is known for its Gothic architecture and stunning stained glass windows.\n\nThese are just a few of the many attractions that Paris has to offer. With so much to see and do, it's no wonder that Paris is one of the most popular tourist destinations in the world."<|end|>
<|user|>
What is so great about #1?<|end|>
<|assistant|>

微调2：

```
git
clone https://github.com/modelscope/swift.git
cd swift
pip install -e .[all]

CUDA_VISIBLE_DEVICES=0,1,2,3
NPROC_PER_NODE=4 \
```

```
swift sft \
--model_type phi3-4b-4k-instruct \
--dataset ms-agent-for-agentfabric-default alpaca-en ms-bench ms-agent-for-agentfabric-addition coig-cqia-ruozhiba coig-cqia-zhihu coig-cqia-exam coig-cqia-chinese-traditional coig-cqia-logi-qa coig-cqia-segmentfault coig-cqia-wiki \
--batch_size 2 \
--max_length 2048 \
--use_loss_scale true \
--gradient_accumulation_steps 16 \
--learning_rate 5e-5 \
--use_flash_attn true \
--eval_steps 500 \
--save_steps 500 \
--train_dataset_sample -1 \
--dataset_test_ratio 0.1 \
--val_dataset_sample 10000 \
--num_train_epochs 2 \
--check_dataset_strategy none \
--gradient_checkpointing true \
--weight_decay 0.01 \
```

```
--warmup_ratio 0.03 \
--save_total_limit 2 \
--logging_steps 10 \
--sft_type lora \
--lora_target_modules ALL \
--lora_rank 8 \
--lora_alpha 32

--custom_train_dataset_path xxx.jsonl \
--custom_val_dataset_path yyy.jsonl \

# Experimental environment: A100
CUDA_VISIBLE_DEVICES=0 \
swift infer \
--ckpt_dir "/path/to/output/phi3-4b-4k-instruct/vx-xxx/checkpoint-xxx" \
--load_dataset_config true \
--max_new_tokens 2048 \
--temperature 0.1 \
--top_p 0.7 \
--repetition_penalty 1. \
--do_sample true \
--merge_lora false \
```

4.4 应用场景

正因为Gemma是在包含多种来源的文本数据集上训练的，总计6万亿个标记，这使模型接触代码有助于其学习编程语言的语法和模式，改善了其生成代码或理解代码相关问题的能力。此外，在数学文本上的训练，在数据准备过程的多个阶段应用严格的CSAM过滤，以确保排除有害和非法内容。作为使Gemma预训练模型安全可靠的一部分，使用自动化技术从训练集中过滤掉某些个人信息和其他敏感数据。也还有其他方法，根据我们的政策过滤基于内容质量和安全的内容。

对于Llama-3而言，Llama还有一个庞大的开源社区，开发团队与GGML团队等也有着密切的合作关系，还包括Yarn项目（能够扩展上下文长度）等各式各样的相关开源项目。Llama相关的公司非常多，包括硬件供应商，如Nvidia、Intel和Qualcomm，还有各种下游企业和平台提供商。团队想要推出更大更好的模型，支持多种语言：Facebook（FOA）的家庭应用程序已经覆盖了近40亿的用户，多语言对于Llama目标实现的AI场景，以及多模态功能都至关

重要，包括在Ray-Ban智能眼镜上实现AI，需要理解周围的一切，不可能仅仅通过文字来实现，所以多模态功能在未来肯定也会推出。

即便是Phi-3这种端侧模型，输入文本字符串，如一个问题、一个提示或要被总结的文档不在话下。输出对输入的英语文本生成响应，如对一个问题的回答或文档的摘要，生成诗歌、剧本、代码、营销文案和电子邮件草稿等创意文本格式。反过来，生成文本库、研究论文或报告的简洁摘要也是个功能，甚至作为聊天机器人和会话AI，为客户服务、虚拟助手或互动应用提供会话界面。

五、结论

5.1 长处

Gemma 从 Gemini 模型计划的许多学习中受益，包括代码、数据、架构、指令微调、来自人类反馈的强化学习以及评估。具体而言， Gemma 模型将为社区带来净收益，因为我们进行了广泛的安全评估和缓解措施；然而，我们承认此版本不可逆转，开放模型造成的危害尚未得到很好的定义，因此我们将继续根

据这些模型的潜在风险采取适当的评估和安全缓解措施。即使在基准任务上表现出色，仍需要进一步研究才能创建鲁棒、安全的模型，可靠地按预期执行。示例进一步研究领域包括事实性、对齐、复杂推理和对抗性输入的鲁棒性。正如 Gemini 技术报告中所讨论的，我们重申了 LLM 使用的非详尽限制列表。正如 Gemini 所讨论的，我们注意到需要更具挑战性和鲁棒性的基准测试。在Gemma禁止使用政策中概述了禁止使用的用途。

5.2 局限与应对方案

虽然投入了大量精力改进模型，但开发者也认识到它的局限性。除了惠及人工智能开发生态系统之外，我们还认识到大型语言模型的恶意使用，例如创建深度伪造图像、人工智能生成的信息以及非法和令人不安的材料，可能会在个人和机构层面造成伤害 (Weidinger 等人，2021 年)。此外，提供模型权重而不是在 API 后面发布模型，对负责任的部署提出了新的挑战。

首先，恶意行为者出于恶意目的微调 Gemma，尽管他们的使用受制于使用条款，该条款禁止以 Gemma 禁止使用政策的方式使用 Gemma 模型。但是需要进一步努力构建更稳健的缓解策略来应对开放系统的故意滥用，谷歌 DeepMind 将继续在内部和与更广泛的人工智能社区合作方面进行探索。面临的第二个挑战是保护开发人员和下游用户免受开放模型的非预期行为，包括生成有害语言或延续歧视性社会危害、模型幻觉和泄露个人可识别信息。在 API 后面部署模型时，可以通过各种过滤方法降低这些风险。

由于 Gemma 系列模型缺少这一层防御，开发者努力按照 Gemini 的方法过滤和测量预训练数据中的偏差，通过标准化的人工智能安全基准评估安全性；加成内部的红队测试，以更好地了解与 Gemma 外部使用相关的风险，最后对模型进行严格的道德和安全评估，评估结果见第 8 为了确保下游用户的透明度，我们发布了详细的模型说明卡，为研究人员提供更全面的了解 Gemma 的信息。我们还发布了生成式人工智能责任工具包，以支持开发人员负责任地构建 AI。总而言之，考虑到现有生态系统中可访问的更大系统的能力，Gemma 的发布将对整体人工智能风险组合产生微不足道的影响。

-

Phi-3 只是在前几个迭代学习的基础上更进一步。Phi-1 专注于编码，Phi-2 开始学习推理，而 Phi-3 则更擅长编码和推理。虽然Phi-3系列模型知道一些常识，但它在广度上无法击败GPT-4或其他LLM--从一个在整个互联网上接受过训练的LLM和一个像Phi-3这样的小型模型中得到的答案差别很大。

在LLM的能力方面，虽然phi-3-mini模型在语言理解和推理能力上与更大的模型相当，但在某些特定任务上，它由于模型规模的限制而存在根本性的局限。例如，在TriviaQA上表现不佳就反映出模型没有足够的容量来存储大量的“事实性知识”。不过，团队认为这种局限可以通过与搜索引擎相结合的方式来克服。例如，我们在图4中展示了一个实例，即在HuggingFace默认的Chat-UI中使用phi-3-mini。

另一个局限是，Phi-3主要将模型的语言限定为英语。对于小型语言模型而言，开发其多语言处理能力是重要的下一步。通过引入更多的多语言数据，phi-3-small已经展现出了一些初步而积极的成果。尽管

团队在负责任人工智能（RAI）方面做出了不懈努力，但与大多数LLM一样仍然面临着事实不准确（或称之为“幻觉”）、重现或放大偏见、生成不适当内容和安全问题的挑战。通过精心策划的训练数据、针对性的后期训练以及借鉴红队的见解所做的改进，在各个方面都显著减少了这些问题。然而，要完全克服这些挑战还有很长的路要走。

六、新模型预告

七、附录・参考文件

https://zhuanlan.zhihu.com/p/683315591
https://www.kaggle.com/models/google/gemma
https://cloud.google.com/dataflow/docs/machine-learning/gemma-run-inference?hl=zh-cn