基于 LLM 实现特定垂直领域的知识问答

方桂安，刘梦莎，梁静蕾

摘 要—在本次期末项目中，我们的团队着重在 NLPCC 2016 KBQA 任务集上微调并比较了 GPT-2、Galactica 和 LLaMA 三种先进的预训练模型。通过创新性地引入多轮对话以及实时的人类反馈机制，我们成功地增强了模型的交互性与适应性。在各项评估指标上，我们的实验结果表明，这些模型都实现了对比 Bert 模型时的显著提升。这些发现明确地表明了针对特定垂直领域的知识问答，基于大规模语言模型（LLM）实现的方法具有显著的优越性。本项目中由梁静蕾负责模型部分，刘梦莎负责算法部分，方桂安负责实验等其他内容。

关键词—LLaMA, LoRA, RLHF, GPT-2, Galactica

I. 实验概述

ChatGPT 掀起了大模型 (LLM) 风潮之后，一大波 LLMs(GPT-4, LLaMA, BLOOM, Alpaca, Vicuna, MPT ..) 百花齐放。知识问答、文章撰写、代码编写和纠错、报告策划等等，它们都会，也能够交互式地和人类玩文字游戏，甚至可以实现将 LLM 作为交互的接口，同时连接到其它各种模态 (e.g. 视觉 & 语音) 的模型，从而创造了炸裂的多模态效果。

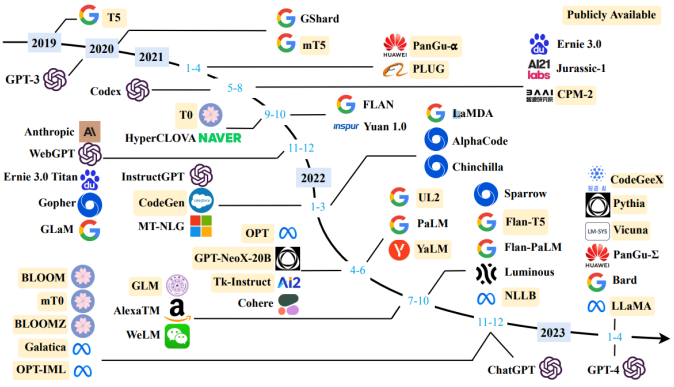


Fig. 1. LLM 时间轴

在期中项目中，我们已经利用所学知识尽可能提升基于知识库的自动问答系统的性能，然而效果并不能让我们满意。所以在本次期末项目中，我们利用 LoRA、RAFT 等技术微调了 LLaMA，并且 finetune 了参数量较少的 GPT-2 和 Galactica 作为 baseline 进行对比，最终实现了一个完善的自动问答 bot。

方桂安，20354027，(e-mail: fanggan@mail2.sysu.edu.cn)。
刘梦莎，20354091，(e-mail: liumsh6@mail2.sysu.edu.cn)。
梁静蕾，20354072，(e-mail: liangjlei@mail2.sysu.edu.cn)。

II. 模型

A. GPT-2

1) Introduction to GPT-2: GPT-2 是 GPT 的一个改进模型，其模型结构与 GPT 相比几乎没有什么变化，只是让模型变得更大更宽，并且取消了 Fine-tuning 的步骤。也就是说 GPT-2 采用了一阶段的模型（预训练）代替了二阶段的模型（预训练 + 微调），并且在语言模型（文本摘要等）相关领域取得了不错的效果。

GPT-2 的核心是语言模型，语言具有天然的顺序性，通常可以表示为：

$$p(x) = \prod_{i=1}^{n} p\left(s_n \mid s_1, \cdots, s_{n-1}\right)$$

可以泛化成：

$$p\left(s_{n-k}, \cdots, s_n \mid s_1, \cdots, s_{n-k-1}\right)$$

针对单个任务可以表示为估计一个条件分布：

$$p(\text{output} \mid \text{input})$$

对于一个通用的系统来说，可以适用于多种任务：

$$p(\text{output} \mid \text{input, task})$$

所以语言模型也能够学习某些监督学习的任务，并且不需要明确具体的监督符号。而监督学习由于数据量的关系通常只是无监督学习的一个子集，所以无监督学习的全局最小也必定是监督学习的全局最小，所以目前的问题变为了无监督学习是否能收敛。

2) Architectural Design of GPT-2: GPT-2 基于 GPT 上基础增加了两个 Layer normalization，一个加在每个 sub-block 输入的地方，另一个加在最后一个 self-attention block 的后面。模型中只使用了多个 Masked

Self-Attention 和 Feed Forward Neural Network 两个模块。同时考虑到模型深度对残差路径的累积问题，GPT-2 采用了修正的初始化方法。在初始化时将残差层的权重缩放到 $\frac{1}{\sqrt{(N)}}$

此外，vocabulary 的大小扩展到了 50257，输入的上下文大小从 512 扩展到了 1024，并且使用更大的 batch size（512）。

GPT-2 提供了四种不同规模的模型：

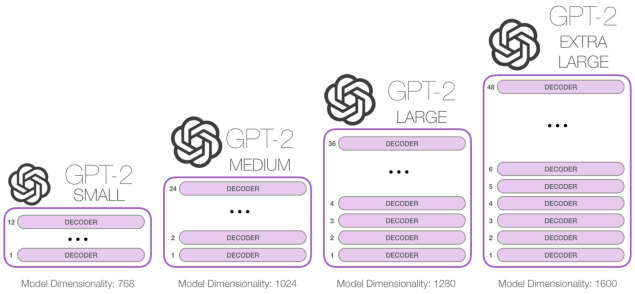


Fig. 2. GPT-2 模型

对于模型的输入而言，分为 Token Embedding 和 Position Embedding。实际的输入由 Position Embedding 和 Token Embedding 相加得到。

Fig. 3. input

将输入送到模型后，首先通过 Masked Self-Attention，然后通过 Feed Forward Neural Network，第一个子模块处理完成后会传送到下一个模块进行的计算，每一个模型处理方式相同，但是各自独立。

每一层的 Self-Attention 都会融入了模型对于用来解释某个单词上文的相关单词的理解。具体做法是，给序列中每一个单词都赋予一个相关度得分，然后对他们的向量表征进行加权求和：

来自 Self-Attention 的输出作为 Feed Forward 的输入，并经过两层神经网络产生输出。将模型最后的输出与 Token Embedding 矩阵相乘，并通过 Logits 回归得到最终的概率值，取最大值即为输出。

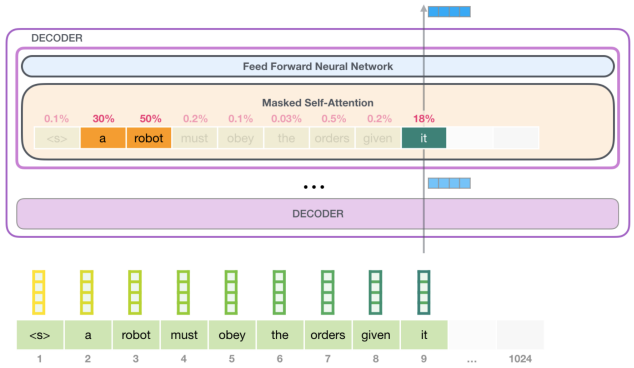


Fig. 4. Weighted Sum Result

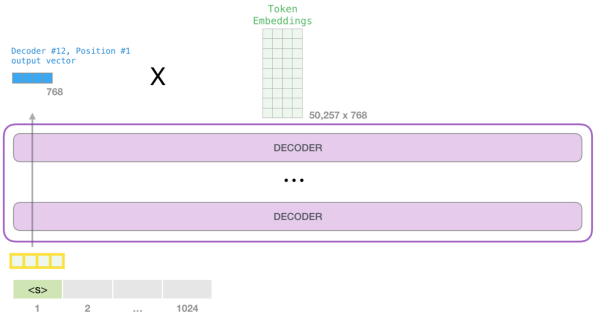


Fig. 5. output

3) Training Methodology: GPT-2 的思想是尽可能使训练数据大且更加多样化，以此来学习到不同领域的知识。

由于 common crawl 获得的数据集往往会存在质量问题，因此在 GPT-2 只爬取人工筛选过的 web 数据。GPT-2 通过爬取 Reddit 上所包含的链接（这里是以一种启发式的方法去获取人工筛选的数据，GPT-2 认为 Reddit 这个社交媒体上挂的链接，可能是人们比较感兴趣或认为有价值的，因此存在数据质量问题的概率比较小）。

最终 GPT-2 获得的数据集为‘WebText’，包含了 4500 万个链接的文本数据。经过清洗和各种处理之后包含 800 万个文档，共计 40GB 的文本数据。另外还去除了维基百科的数据，以防和论文中使用的其它数据集存在重复。

数据预处理方面 GPT-2 综合考虑了 OOV 问题和词表过大的问题，使用了 BPE 算法，这能够使模型处理任何字符。

B. Galactica

1) Introduction to Galactica: 信息过载是科学进步的一个主要障碍。科学文献和数据的爆炸性增长使得在大量的信息中发现有用的见解变得越来越难。人们通过搜索引擎获取科学知识，但单靠它们无法组织科学知识。因此 Galactica 被设计用来帮助梳理科学信息，帮助研究界更好地管理“信息爆炸”。

2) Architectural Design of Galactic: Galactic 的解码器设置中使用了 Transformer 架构，并进行了以下修改：

1) GeLU 激活：对所有模型的尺寸都使用了 GeLU 激活；
2) 上下文窗口：对所有的模型尺寸都使用了 2048 长度的上下文窗口；
3) 无偏差；遵循 PaLM ，不在任何密集核或层规范中使用偏差；
4) 学习的位置嵌入：对模型使用学习的位置嵌入，在较小的尺度上试验了注意力加上线性偏置的方法，但没有观察到大的收益，所以研发者们没有使用它；
5) 词语：使用 BPE 构建了一个包含 50k 个标记组成的词汇表，词汇量是由随机选择的 2% 的训练数据子集中产生的。

Galatcica 团队总共训练了五个模型，参数在 1.25 亿到 1200 亿之间。据该团队称，Galactica 的性能随着规模的扩大而平稳地增加。此次期末作业中我们用的是 1.3B 的模型。

Model	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{heads}	Batch Size	Max LR	Warmup
GAL 125M	125M	12	768	12	64	0.5M	6×10^{-4}	375M
GAL 1.3B	1.3B	24	2,048	32	64	1.0M	2×10^{-4}	375M
GAL 6.7B	6.7B	32	4,096	32	128	2.0M	1.2×10^{-4}	375M
GAL 30B	30.0B	48	7,168	56	128	2.0M	1×10^{-4}	375M
GAL 120B	120.0B	96	10,240	80	128	2.0M	0.7×10^{-5}	1.125 B

TABLE I
MODEL

3) Training Methodology: Galactica 用 4800 万篇论文、教科书和讲义、数以百万计的化合物和蛋白质、科学网站、百科全书以及来自”自然书”数据集的更多内容进行了训练。

鉴于存在的不同模式，标记化是数据集设计的重要组成部分。例如，蛋白质序列是根据氨基酸残基编写的，其中基于字符的标记化是合适的。为了实现 specialized tokenization 的目标，Galactica 为不同的模式使用专用 tokens，如表 III 所示：

Total dataset size = 106 billion tokens			
Data source	Documents	Tokens	Token %
Papers	48 million	88 billion	83.0%
Code	2 million	7 billion	6.9%
Reference Material	8 million	7 billion	6.5%
Knowledge Bases	2 million	2 billion	2.0%
Filtered CommonCrawl	0.9 million	1 billion	1.0%
Prompts	1.3 million	0.4 billion	0.3%
Other	0.02 million	0.2 billion	0.2%

TABLE II
GALACTICA 数据集来源

Modality	Entity	Sequence
Text	Abell 370	Abell 370 is a cluster...
LATEX	Schwarzschild radius	$r_{-}\{s\} = \backslash f$ rac{2GM} {c^2}
Code	Transformer	class Transformer(nn.Module)
SMILES	Glycine	$C(C(=0)0)N$
AA Sequence	Collagen $\alpha-1$(II) chain	MIRLGAPQTL . .
DNA Sequence	Human genome	CGGTACCCTC..

TABLE III
SPECIALIZED TOKENS

Galactica 选择用更多的任务提示来增强预训练数据，以提高低尺度下的性能。消除了对更多数据规模的需求，或更多的模型规模。

模型使用 metaseq 库训练，该库由 Meta AI 的 NextSys 团队构建。如图 6 通过四个训练阶段，所有模型大小的验证损失都在下降。为了训练最大的 120B 模

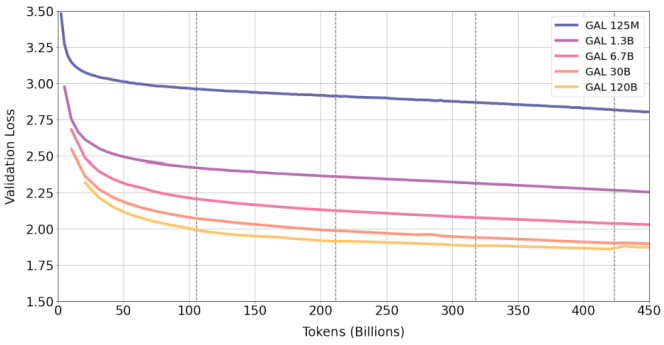


Fig. 6. Repeated Tokens and Validation Loss

型，Galactic 使用 128 个 NVIDIA A100 80GB 节点。

4) Performance and Use-Cases: Galactic 应对技术知识测试的表现，如 LaTeX 方程式，Galactica 以 68.2% 对 49.0% 的优势胜过 GPT-3。Galactica 在回答生物学和医学（PubMedQA 和 MedMCQA）的技术问题方面也取得了新的记录（77.6% 和 52.9%）。与其他大型语言模型相比，Galactica 产生的有毒内容较少。

作为具体的应用场景，Galactica 可以自己归纳总

结创建文献综述、维基文章或科学主题的讲义，或回答包括引文在内的科学问题。

C. LLaMA

1) Introduction to LLaMA: LLaMA 是一个参数范围从 7B 到 65B 的基础语言模型集合。LLaMA 仅仅在公开的数据集上进行训练了多个尺度的模型，就可以达到最先进的效果。且对模型和实现方式进行优化，加速训练。LLaMA-13B 比 GPT-3 在大多数 benchmarks 上性能都要好，但是模型大小缩减到十分之一。

2) Architectural Design of LLaMA: 从模型的结构上来说，LLaMA 使用了这些大模型改进技术:

1) 使用 pre-normalization（GPT-3）来提高模型训练的稳定性
2) 使用 SwiGLU 激活函数替代 ReLU（PaLM）
3) 使用 Rotary Embeddings 来替代绝对位置 embeddings（GPTNeo）

其各尺寸 LLaMA 模型的超参数如表 IV 所示，此次实验我们选用的模型参数是 7B。

params	dimension	n heads	n layers	learning rate	batch size	n tokens
6.7B	4096	32	32	$3.0e^{-4}$	4M	1.0 T
13.0 B	5120	40	40	$3.0e^{-4}$	4M	1.0 T
32.5B	6656	52	60	$1.5e^{-4}$	4M	1.4 T
65.2B	8192	64	80	$1.5e^{-4}$	4M	1.4 T

TABLE IV
LLaMA 超参数

3) Train Metholodogy: LLaMA 的训练数据集是由几个来源的混合组成，如表 V，涵盖了各种领域。并给出了相应的占比:

Dataset	Sampling prop.	Epochs	Disk size
CommonCrawl	67.0%	1.10	3.3 TB
C4	15.0%	1.06	783 GB
Github	4.5%	0.64	328 GB
Wikipedia	4.5%	2.45	83 GB
Books	4.5%	2.23	85 GB
ArXiv	2.5%	1.06	92 GB
StackExchange	2.0%	1.03	78 GB

TABLE V
LLaMA 数据集

LLaMA 使用标准优化器在大量文本数据上训练大型转换器。LLaMA 使用 AdamW 优化器，超参数是 $\beta_1 = 0.9, \beta_2 = 0.95$ ，使用余弦学习率调度，这样最终的学习率是最大的学习率的十分之一，使用的权重衰减为 0.1 ，梯度剪裁为 1 ，使用 2000 个预热步骤，并且根据模型的大小而改变学习率和批处理大小。

此外，LLaMA 通过使用因果多头注意力来减少内存的使用和运行时间。

4) Performance and Use-Cases: LLaMA 团队使用 zero-shot 和 few-shot 的任务，在 20 个 benchmarks 上进行实验。

主要实验的内容有:

1) **常识推理**

在表 VI 中，作者与现有的各种规模的模型进行比较，并报告了相应论文中的数字。首先，LLaMA-65B 在所有报告的基准上都超过了 Chinchilla-70B，除了 BoolQ。同样，除了在 BoolQ 和 WinoGrande 上，这个模型在任何地方都超过了 PaLM540B。LLaMA-13B 模型在大多数基准上也超过了 GPT-3，尽管其体积小了 10 倍。

		BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA
GPT-3	175B	60.5	81.0	-	78.9	70.2	68.8	51.4	57.6
Gopher	280 B	79.3	81.8	50.6	79.2	70.1	-	-	-
Chinchilla	70 B	83.7	81.8	51.3	80.8	74.9	-	-	-
PaLM	62B	84.8	80.5	-	79.7	77.0	75.2	52.5	50.4
PaLM-cont	62B	83.9	81.4	-	80.6	77.0	-	-	-
PaLM	540 B	88.0	82.3	-	83.4	81.1	76.6	53.0	53.4
LLaMA	7 B	76.5	79.8	48.9	76.1	70.1	72.8	47.6	57.2
	13B	78.1	80.1	50.4	79.2	73.0	74.8	52.7	56.4
	33B	83.1	82.3	50.4	82.8	76.0	80.0	57.8	58.6
	65B	85.3	82.8	52.3	84.2	77.0	78.9	56.0	60.2

TABLE VI
LLaMA 与其他大模型数据

2) **闭卷问答**

在表 VII 中，作者比较了 LLaMA 和其他模型在 NaturalQuestions 基准上的性能。LLaMA-65B 在 0-sot 和 few-shot 设置中都达到了最先进的性能。更重要的是，尽管 LLaMA-13B 比 GPT-3 和 Chinchilla 小 5-10 倍，但在这些基准测试中也具有竞争力。该模型在推理过程中在单个 V100 GPU 上运行。

在大多数任务中，LLaMA-13B 要比 GPT-3(175B) 的性能要好，LLaMA-65B 和模型 Chinchilla-70B 以及 PaLM-540B 的实力相当。

III. 算法

A. LoRA

1) 背景: 自大模型 (LLM) 风潮开启之后，成百上千亿参数量的模型让普通人望而却步。而 LoRA 的出现极大地减少了需要的显存 (GPU) 资源。LoRA 的英文全

		0-shot	1-shot	5-shot	64-shot
GPT-3	175B	14.6	23.0	-	29.9
Gopher	280B	10.1	-	24.5	28.2
Chinchilla	70B	16.6	-	31.5	35.5
PaLM	8B	8.4	10.6	-	14.6
	62B	18.1	26.5	-	27.6
	540B	21.2	29.3	-	39.6
LLaMA	7B	16.8	18.7	22.0	26.1
	13B	20.1	23.4	28.1	31.9
	33B	**24.9**	28.3	32.9	36.0
	65B	23.8	**31.0**	**35.0**	**39.9**

TABLE VII
LLaMA 与其他大模型数据

称是 Low-Rank Adaptation of Large Language Models，直译为大语言模型的低阶适应，是一种 PEFT（参数高效性微调方法），这是微软的研究人员为了解决大语言模型微调而开发的一项技术，比起全量 fine-tune ，它的可训练的参数量少了很多。

在 LoRA 诞生之前，比较有代表性的 PEFT 方法主要有额外引入适配下游任务的 adapter layers 和优化靠近模型输入层的激活 (activations) 两种。对于后者，比较有代表性的有 prefix tuning 等。

对于引入 adapter layers 这类方法，它们的不足在于：新增的 adapter layers 必须串行处理，从而增加了推理延时；在分布式训练时需要更多的 GPU 同步操作 (如 All-Reduce & Broadcast)。而另一类方法，以 prefix tuning 为例，它们本身很难优化，且这种方法需要在模型的输入序列中预留一部分用作可微调的 prompt，从而限制了原始输入文本的句长。

LoRA 这项用于微调 LLMs 的技术额外引入了可训练的低秩分解矩阵，同时固定住预训练权重。它的重点在于：预训练权重不需训练，因此没有梯度，仅训练低秩矩阵那部分的参数。LoRA 通过优化秩分解矩阵来间接训练一些密集层，从而降低大型语言模型的计算和存储成本。

2) **方法概述**: LoRA（Low-Rank Adaptation）的基本思想源于对过参数化模型的研究。这些研究表明，模型在适配下游任务后，权重矩阵的本征秩变得很低，这就代表着其实并不需要这么高的维度数就能够进行表征了，在高维度的权重矩阵中存在冗余。因此，LoRA 的设计假设在模型在微调过程中，权重更新的那部分 (参数矩阵) 肯定也是低秩 (low rank) 的。而权重更新的部分可以表示为 $W = W_0 + \Delta W$ 。其中 W_0 就是更新前的权重 (在一开始就是预训练权重)，而 ΔW 就是更新的那部分，也就是经过反向传播得到梯度后计算出来的需要更新的量。

假设 $W \in R^{d\times k}$ ，由于梯度与权重参数是一对一的，因此 $\Delta W \in R^{d\times k}$ 。如今，既然认为 ΔW 的本征秩很低，那么不妨对其做低秩分解: $\Delta W = BA$, 其中 $B \in R^{d\times r}, A \in R^{r\times k}$ ，并且 $r \ll \min(d,k)$ 。这个 r 就是所谓的 low rank，因为它远小于 d 和 k 。由此可知, 经过低秩分解后, $\Delta W = BA$ 这部分的参数量是远小于预训练权重 W 的。按理来说, ΔW 是在反向传播阶段才会出现的, 但我们可以将其 “提前拿出来”：让它和 W_0 一起参与前向过程。这样一来，反向传播时梯度就只会传导到 $\Delta W = BA$ 这部分，因其本身就是待更新量, 而 W_0 初始是预训练权重, 被固定了, 无需接收梯度。

经过 LoRA 的 “洗礼”，现在如果你喂给模型一个输入 x ，前向过程就会表示为:

$$Wx = W_0x + \Delta Wx = W_0x + BAx(i)$$

由于 $\Delta W = BA$ 的参数量远小于 W, 因此, 相比于全量 fine-tune , LoRA 降低了 optimizer states 这部分对于显存资源的需求。这是因为 optimizer 对于需要更新的模型参数会保存一份副本, 在全量 fine-tune 下, W 要全量更新, 于是 optimizer 保存的副本参数量为 dxk ; 而 LoRA 仅需更新 $\Delta W = BA$ 这部分, 所以 optimizer 保存的副本参数量仅为 $dxr + rxk$, 其中 r 是远小于 d, k 的。

3) **工作原理**: LoRA 通过以下方式应用低秩适应方法来降低大型语言模型的计算和存储成本：

1) **选择应用 LoRA 的权重矩阵**：在 Transformer 架构中，自注意力模块有四个权重矩阵 (W_q, W_k, W_v, W_0)，MLP 模块有两个。LoRA 作者选择只对注意力权重进行适应，冻结 MLP 模块，这样做既简单又参数高效。他们进一步研究了适应 Transformer 中不同类型的注意力权重矩阵的效果。
2) **低秩分解**：对于预训练的权重矩阵 W0，LoRA 通过低秩分解来约束其更新，即 $W_0x + \Delta Wx = W_0x + BA$，其中 B 和 A 是可训练的参数，r 是 LoRA 模块的秩。在训练过程中，W_0 被冻结并且不接收梯度更新，而 A 和 B 包含可训练的参数。
3) **优化和推理**：在训练过程中，我们不需要为大部分参数计算梯度或维护优化器状态，而只需要优

化注入的、规模较小的低秩矩阵。在部署时，我们可以将可训练的矩阵与冻结的权重合并，不会引入额外的推理延迟。这些矩阵可以学习新任务的特定知识，而不会改变预训练模型的权重。

4) **任务切换**：LoRA 在适应过程中不改变预训练模型的权重，预训练模型可以共享并用于为不同任务构建许多小的 LoRA 模块。并通过替换图 7中的矩阵 A 和 B 来有效地切换任务，从而大大减少存储需求和任务切换开销。LoRA 在适应过程中不改变预训练模型的权重。这意味着预训练模型的知识被保留，而不会在适应过程中被覆盖或遗忘。

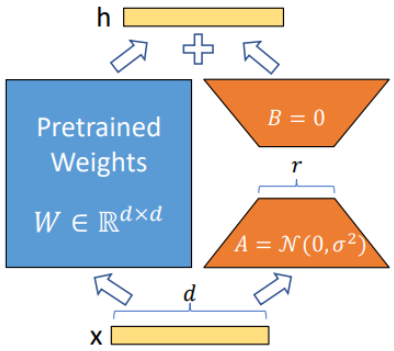


Fig. 7. reparametrization 过程

这种方法大大减少了下游任务的可训练参数数量。与全面微调相比，LoRA 可以将可训练参数的数量减少 10000 倍，GPU 内存需求减少 3 倍。尽管 LoRA 具有更少的可训练参数，更高的训练吞吐量，并且没有额外的推理延迟，但其性能与全面微调相当，甚至更好。

4) LORA 方法的优点:

- 在面对不同的下游任务时，仅需训练参数量很少的低秩矩阵，而预训练权重可以在这些任务之间共享。
- 省去了预训练权重的梯度和相关的 optimizer states，大大增加了训练效率并降低了硬件要求。
- 训练好的低秩矩阵可以合并 (merge) 到预训练权重中，多分支结构变为单分支，从而达到没有推理延时的效果。
- 与之前的一些参数高效的微调方法 (如 Adapter, Prefix-Tuning 等) 互不影响，并且可以相互结合。

5) LORA 方法的局限性:

- 批处理输入的问题：如果选择将 A 和 B 吸收到 W 中以消除额外的推理延迟，那么在单个前向传递中批处理不同任务的输入就不是直接的。
- 适应性的限制：虽然理论上可以将 LoRA 应用到神经网络中的任何子集的权重矩阵，以减少可训练参数的数量，但在实践中，作者选择了只适应注意力权重，冻结 MLP 模块，但对于适应 MLP 层、LayerNorm 层和偏置的实证研究，还需要进一步的工作。

B. RAFT

1) 背景: 生成基础模型容易受到隐式偏见的影响，这些偏见可能源自大量的无监督训练数据，导致带有偏见的和不公平的结果，而这可能产生严重的后果。因此，将这些模型与人类的伦理和偏好对齐是一个重要步骤。

为了解决这个问题，预训练模型通常在下游任务上进行微调，使用定制的数据，要么是为了在专门的任务中提高性能，要么是为了消除原始模型中的潜在偏见。一种方法是使用标记数据以监督的方式微调预训练模型，这被称为监督微调（SFT）。然而，收集新的监督样本在实际应用中可能会很昂贵，尤其是需要专家参与生成高质量数据的情况。最近，人类反馈的强化学习（RLHF）已经成为微调预训练生成模型的一种有效的方法。在最近的 LLMs 研究中，RLHF 已经被广泛用于使用基于策略的深度强化学习（DRL）算法微调预训练模型，这些算法通常是近端策略优化（PPO）。RLHF 的想法是通过优化反映特定人类偏好（例如，道德，无害）的奖励函数，使语言模型与人类偏好和社会价值观对齐。例如，OpenAI 使用强调某些人类价值观的奖励函数，使用 RLHF 微调了 GPT-3 的一个版本。然而，与 RL 算法相关的低效率和不稳定性经常成为成功对齐生成模型的重大障碍，因此需要开发更强大、更简洁的方法。

而 RAFT 为 AIGC 社区的研究者和工作者提供了一种新的可选的 AI 对齐策略。香港大学提出了一个对齐框架，RAFT (Reward rAnked FineTuning)，它使用奖励函数对生成模型的输出进行排名，然后继续使用选定的样本进行 SFT-like 的训练。这种方法鼓励生成模型优先考虑奖励较高的样本。与 PPO 相比，我 RAFT 方法提供了显著的计算优势，节省了大量的内存和梯度计算。此外，由于 SFT-like 训练的稳定性，这种方法表现出较低的样本复杂性，并需要较少的可调参数，可以应用于任何生成模型。

相较于 OpenAI 所用 RLHF 对齐算法的高门槛，

RAFT(Reward rAnked Fine-Tuning) 易于实现，在训练过程中具有较高的稳定性，并能取得更好的对齐效果。任意生成模型都可以用此算法高效对齐，NLP/CV 通用。将其用在 Stable Diffusion 上，还能对齐生成图片和提示词，让模型生成更加符合提示词描述的图片。

2) 算法流程: OpenAI 在 ChatGPT 前身 Instruct 论文中介绍了基于人类反馈的强化学习 (RLHF) 算法。首先利用人类标注数据训练一个打分器 (reward model)，然后通过强化学习算法（如 PPO）来调节模型的行为，使得模型可以学习人类的反馈。但 PPO 等强化学习算法高度依赖反向梯度计算，导致训练代价较高，并且由于强化学习通常具有较多的超参数, 导致其训练过程具有较高的不稳定性。

相比之下，RAFT 算法的基本思想是利用奖励函数对生成模型的输出进行排名，然后选择性地对这些输出进行监督式微调（SFT）。这种方法鼓励生成模型优先考虑奖励较高的样本，从而基于这些样本微调一个对人类更友好的 AI 模型。

具体而言，RAFT 分为四个核心步骤：

1) **数据收集**：数据收集可以利用正在训练的生成模型作为生成器，也可以利用预训练模型（例如 LLaMA、ChatGPT，甚至人类）和训练模型的混合模型作为生成器，有利于提升数据生成的多样性和质量。
2) **计算奖励**：对于每个输入- 响应对（x，y），计算奖励 r(x, y)。这个奖励是由任务特定的奖励函数计算出来的，它可以为每个输入- 响应对分配一个奖励信号。
3) **数据排序**：如果 r(x, y) 大于 B 中最小的奖励，那么就将 (x, y) 和 r(x, y) 插入到 B 中，并移除 B 中奖励最小的样本。这样，B 始终包含了最大的奖励样本，这些样本将用于后续的微调。
4) **模型微调**：最后，对 B 中的样本进行微调，得到新的生成模型 Gt，训练之后的模型能够与人类需求相匹配。这个微调过程使用了 AdamW 优化器，并且可以在任何生成模型上进行。

在 RAFT 算法中，模型利用了更多次采样 (当下采样后用以精调的样本一定时)，和更少次梯度计算（因为大部分低质量数据被 reward 函数筛选掉了），让模型更加稳定和鲁棒。同时，在某些情况下，由于有监督微调本身对于超参数敏感性更低，有更稳健的收敛性，在相同 reward 情况下，RAFT 可以拥有更好的困惑度 (perplexity, 对应其生成多样性和流畅性更好)。另一个重要特性是，它可以在任何生成模型上进行，只要计算资源和内存资源允许在该模型上进行 SFT。此外，由于数据收集和模型微调是分开进行的，因此可以使用批量推理和模型并行化来加速数据收集。

Algorithm 1: Reward rAnked FineTuning.

Input: Prompt set $\mathcal{X} = \{x_1, \ldots, x_n\}$, reward function $r(\cdot)$, initial model $G_0 = g(w_0, \cdot)$, acceptance ratio $1/k$, batch size b, temperature parameter α.

for *Stage* $t = 1, \ldots, T$ **do**

1. ***Data collection***. Sample a batch $\mathcal{D}_t$ from $\mathcal{X}$ of size b
   for $x \in \mathcal{D}_t$ **do**
      Generate $y \sim p^{\alpha}_{G_{t-1}}$ and compute $r(x, y)$.
2. ***Data ranking***. Let $\mathcal{B}$ be the $\lfloor b/k \rfloor$ samples with maximum rewards;
3. ***Model fine-tuning***. Fine-tune w_{t-1} on $\mathcal{B}$ to obtain $G_t = g(w_t, \cdot)$.

3) 实验结果: 在实验部分，作者使用了三个任务来测试语言模型，包括电影评论完成、日常对话和文生图。

给出一个电影评论的起始句，RAFT 微调后的大模型可以轻松补齐电影评论，而且更加积极和流畅。在基于 Vicuna 制作的一个心理陪伴机器人演示中，作者模拟了一个因为考试失利而心情低落的人和机器人在聊天。在使用 RAFT 进行对齐之前，模型说自己没有情感和感情，拒绝和人类交友。但是在 RAFT 对齐之后，模型的共情能力明显增强，不断地在安慰人类说，“虽然我是一个 AI，但是我会尽力做你的朋友”。

除了在语言模型上的对齐能力以外，作者还在扩散模型上验证了文生图的对齐能力，这是之前 PPO 算法无法做到的事情。原始 Stable Diffusion 在 256x256 分辨率生成中效果不佳，但经过 RAFT 微调之后不仅产生不错的效果，所需要的时间也仅为原版的 20%。除了提升 256 分辨率图片的生成能力以外，RAFT 还能够对齐生成图片和提示词，让模型生成更加符合提示词描述的图片。

IV. 实验

A. 数据集

1) **数据介绍**: 承接自期中实验，此次项目的数据集依旧是来自 NLPCC ICCPOL 2016 KBQA 任务集，其包含 14,609 个问答对的训练集和包含 9870 个问答对的测试集。并提供一个知识库，包含 6,502,738 个实体、587,875 个属性以及 43,063,796 个三元组。

2) **数据准备**: 在实验开始之前，首先需要对我们的数据集进行预处理和格式转换。由于我们的模型基于文本对文本（text2text）的训练模式，所以我们需要将原始的 KBQA 数据集转换为这种格式。具体地，我们将每个问答对作为一个独立的实例，其中问题作为输入，对应的答案作为输出。

数据准备完成后，我们将得到以下格式的数据：

```
{
  "type": "text2text",
  "instances": [
    {
      "input": "你知道计算机应用基础这本书的作者是谁吗？",
      "output": "秦婉，王蓉"
    },
    {
      "input": "计算机应用基础这本书的出版社是那个？",
      "output": "机械工业出版社"
    },
    ...
  ]
}
```

其中，“input” 是问题文本，“output” 是对应的答案文本。这样的格式可以方便我们直接输入到模型中进行训练

B. 评价指标

在自然语言处理中，评估自动问答（Automatic Question Answering）系统的性能有多种评价指标。下面我们将介绍四个主要的评价指标：MRR、MAP、BLEU 和 ROUGE。

1) MRR (Mean Reciprocal Rank) : MRR 是一种常用的用来评估产生多个答案的问答系统的指标。MRR 是所有答案倒数排名的平均值。具体地，对于一个问题，如果正确答案是系统生成的第一个答案，那么倒数排名就是 1；如果正确答案是第二个答案，那么倒数排名就是 0.5，以此类推。MRR 的公式如下：

$$\mathrm{MRR} = \frac{1}{Q}\sum_{i=1}^{Q}\frac{1}{\mathrm{rank}_i}$$

其中，Q 是问题的总数，rank_i 是第 i 个问题正确答案的排名。

2) MAP (Mean Average Precision) : MAP 是一个用于评估信息检索系统和相关任务的指标，包括问答系统。MAP 适合于在结果列表中可能有多个正确答案的情况，它结合了精确率和召回率的概念。具体地，对于每个查询，我们先计算出一个 AP（Average Precision）值，然后再取所有查询的 AP 值的平均值，即为 MAP。AP 的公式如下：

$$\mathrm{AP} = \frac{1}{m}\sum_{k=1}^{n} P(k)\cdot rel(k)$$

其中，m 是相关答案的总数，n 是返回答案的总数，$P(k)$ 是在前 k 个答案中正确答案的比例，$rel(k)$ 是一个指示函数，如果第 k 个答案是正确的，则 $rel(k)=1$，否则 $rel(k)=0$。然后，MAP 的公式为：

$$\mathrm{MAP} = \frac{1}{Q}\sum_{i=1}^{Q}\mathrm{AP}_i$$

其中，Q 是问题的总数，AP_i 是第 i 个问题的 AP 值。

3) BLEU (Bilingual Evaluation Understudy) : BLEU 是一种用于评估机器翻译系统的指标，但也被用于评估生成式问答系统的性能。BLEU 通过比较系统生成的答案和人类编写的参考答案之间的词匹配来工作。然而，BLEU 分数无法完全反映出语义的准确性和流畅性。BLEU 的计算涉及到精确度 p_n 和 brevity penalty BP，其公式如下：

$$\mathrm{BLEU} = BP\cdot\exp\left(\frac{1}{N}\sum_{n=1}^{N}\log p_n\right)$$

其中，p_n 是 n-gram 精确度，N 是考虑的 n-gram 的最大长度，BP 是简洁性罚项，用于防止系统生成过短的答案。

4) ROUGE（Recall-Oriented Understudy for Gisting Evaluation）： ROUGE 是另一种评估自动摘要和问答系统的指标。ROUGE 主要通过比较系统生成的答案和参考答案之间的 n-gram 匹配度来工作。ROUGE 有多种版本，例如 ROUGE-N、ROUGE-L 和 ROUGE-S 等。其中，ROUGE-N 的公式为：

$$\text{ROUGE-N} = \frac{\sum_{s \in \text{Ref}} \sum_{\text{gram}_n \in s} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{s \in \text{Ref}} \sum_{\text{gram}_n \in s} \text{Count}(\text{gram}_n)}$$

其中，$\text{Count}_{\text{match}}(\text{gram}_n)$ 是 n-gram 在参考答案和系统生成的答案中都出现的次数，$\text{Count}(\text{gram}_n)$ 是 n-gram 在参考答案中出现的次数。

以上这些指标中的每一个都有其局限性，因此在评估问答系统时，通常会结合多个指标使用。例如，MRR 和 MAP 更关注于答案的排序质量，而 BLEU 和 ROUGE 则更注重答案的文本质量。此外，这些指标也可以帮助我们从不同的角度理解问答系统的性能和限制。

C. 实验设置

我们的实验是使用 Pytorch 实现的。所有实验都在配备了 4 块 Nvidia A100 GPU（40GB）显卡和一块 AMD EPYC 7742 CPU（2.30GHz）的服务器上进行。

在实验过程中，我们使用 Weights & Biases 工具进行实验数据的监控和记录。Weights & Biases 是一种专门为深度学习设计的实验跟踪工具，它可以自动记录实验过程中的各种指标，如损失函数值、准确率等，并生成可视化的报告，帮助我们更好地理解和优化模型的训练过程。

最后，为了评估我们的问答系统在实际应用中的表现，我们还自己编写了一个 web 端的问答系统进行测试。用户可以通过这个系统向我们的模型提问，并立即得到模型生成的答案。这样，我们不仅能够对模型的性能进行定量的评估，也能直观地观察模型在处理各种问题时的表现。

D. 奖励监督

我们使用的是香港科技大学的 LMFlow 框架，而不是 Hugging Face 自己维护的 TRL 库。其全流程如图 8 所示。

我们首先按照 InstructGPT 论文的过程，使用 HH-RLHF 数据集训练一个奖励模型，其中包括：监督微调 (SFT) 和通过比较数据集进行奖励建模。由于 PPO 有较大的内存压力，后续实验证明在这个例子的设置里，TRL 的实现无法同时载入 7B 的 RM 与 7B 的训练模型，因此我们选择使用 GPT-Neo-2.7B 作为我们的 RM。

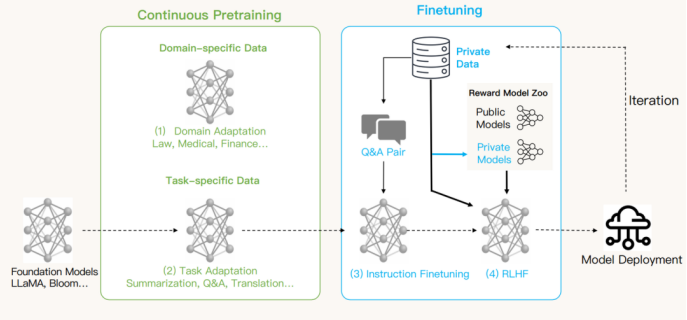


Fig. 8. LMFlow 框架示意图

奖励建模涉及根据给定提示对回应进行排序，通常需要人类标注。幸运的是，LMFlow 开源了一个标注好的数据集，每个提示包含两个回应，包括两个 key: "positive" 和"negative"，其中前者是首选回应。我们默认使用 LoRA 进行训练，因为我们在多个模型上进行了对比，LoRA 由于优化上的优势能节省大量的时间，并且 full training 得到的模型并不能提供明显更有优势的 RM。

RAFT 想法的起源如下，之前有很多研究都发现了如果训练 RM 的数据集直接做 SFT，效果不如先去训练 RM，再用 RL 进行 reward learning。一个解释是后者能够有更多的数据进行训练，但我们注意到前向产生数据本身并不仅仅是 PPO 专属的。此外，当时我们花了很多的时间去调 PPO，发现 PPO 进行训练有容易 OOM，不稳定，模型效果不确定等一些问题，另外就是我们认为在垂直领域 SFT 可以稳定地给模型带来很大的性能提升，一个自然的想法就是，reward learning 是否可以使用 SFT，最终获得的模型在奖励和多样性度量方面都表现良好。

E. 训练过程

在训练过程中，我们使用了三个不同规模的模型：1.3B 参数的 Galactica，1.5B 参数的 GPT-2，以及 7B 参数的 LLaMA。如图 9 所示是三个模型在训练过程中的 loss 变化曲线。其中同一曲线出现不同颜色是因为训练因故打断之后在最新的 checkpoint 上恢复的结果。

1.3B 参数的 Galactica 模型在训练的 10k 步时收敛，损失函数值在此时收敛到 0。Galactica 模型在三个

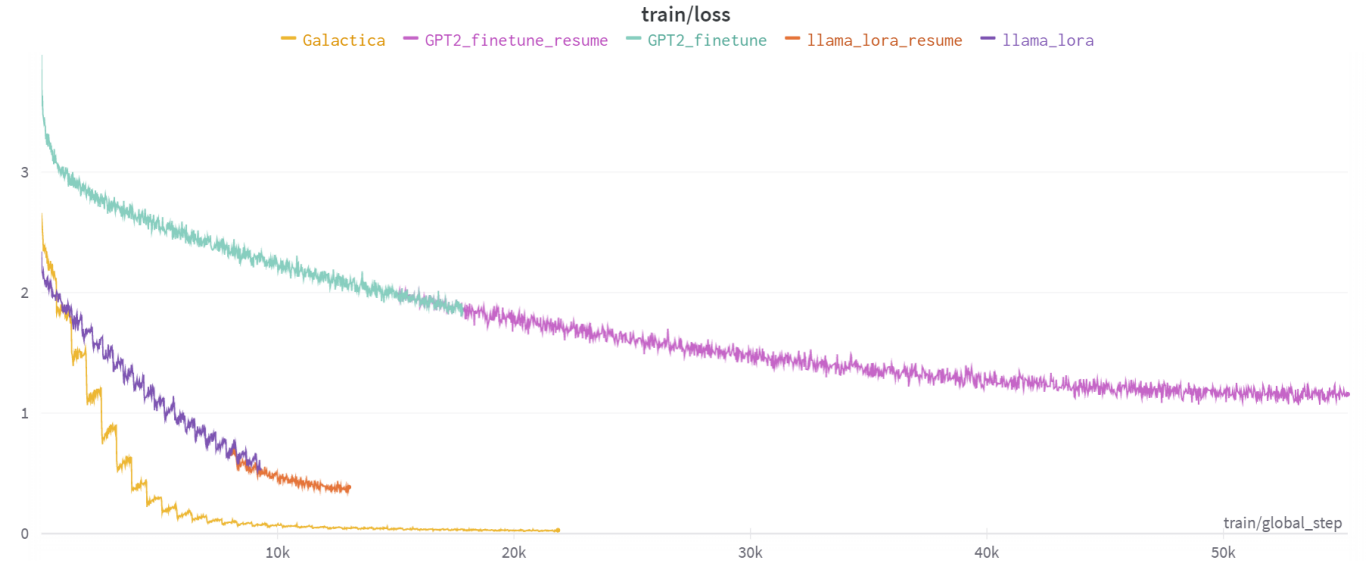


Fig. 9. **模型损失变化图**

模型中参数规模最小，但是它的表现却非常优秀，它需要的训练步骤最少，且最终的损失函数值也最低，这说明 Galactica 在处理我们的任务时具有较高的效率和准确度。

1.5B 参数的 GPT-2 模型的训练在 50k 步时收敛，此时的损失函数值约为 1.5。GPT-2 模型的参数规模中等，但需要更多的训练步骤才能达到收敛，且最终的损失函数值也较高，这可能表明 GPT-2 模型在处理我们的任务时，虽然可以得到可接受的效果，但可能需要更多的优化和调整。

7B 参数的 LLaMA 模型的训练在 13k 步时收敛，此时的损失函数值约为 0.3。LLaMA 模型是三个模型中参数规模最大的，尽管它需要的训练步骤多于 Galactica，但比 GPT-2 少，且最终的损失函数值低于 GPT-2，表明 LLaMA 模型在处理我们的任务时表现出色。

这三个模型的训练过程揭示了模型参数规模对训练效率和模型性能的影响，也展示了在不同的任务和数据集上，可能需要选择不同规模和复杂度的模型才能取得最好的效果。针对该结果，我们推测 Galactica 在该数据集上的自动回答性能最优，GPT-2 最差，而 LLaMA 加载 LoRA 权重之后能取得较好的性能并保持良好的泛化性。

F. 实验结果

1) **定量结果**: 为了全面地了解各个模型在我们的数据集中的表现，我们选择了四个模型进行对比，包括三个预训练的语言模型：1.3B 参数的 Galactica，1.5B 参数的 GPT-2，7B 参数的 LLaMA，以及上一次作业中的 BERT 模型。我们使用 MRR、MAP、BLEU 和 ROUGE 这四种指标来评估模型的性能。

模型	MRR	MAP	BLEU	ROUGE
Galactica (1.3B)	**0.85**	**0.81**	0.78	0.76
GPT-2 (1.5B)	0.65	0.63	0.68	0.67
LLaMA (7B)	0.75	0.73	**0.80**	**0.79**
BERT	0.70	0.68	0.72	0.71

TABLE VIII
各模型的性能

从表格中可以看出，不同模型在各项评估指标上的表现存在一定的差异。具体地，Galactica 模型在 MRR 和 MAP 指标上表现优异，这说明它在处理我们的任务时具有较高的效率和准确度；而 LLaMA 模型在 BLEU 和 ROUGE 指标上的表现更好，这表明该模型在生成质量和信息覆盖度上有优秀的表现。另一方面，BERT 和 GPT-2 的性能稍弱，其中 GPT-2 的性能最低，这可能是由于模型的复杂度较高，导致在处理我们的任务时需要更多的训练时间。

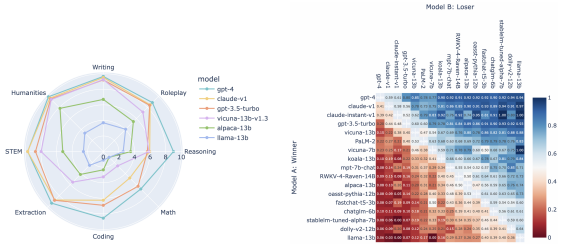
从这些结果中，我们可以看出，尽管 BERT 模型在许多 NLP 任务中都有出色的表现，但在本任务中，其性能仍然无法超越 Galactica 和 LLaMA。这可能是因为，相比于 BERT，Galactica 和 LLaMA 模型更擅长生成任务，更能从给定的上下文中生成连贯、自然的回答。而对于 GPT-2，虽然它的模型参数多于 Galactica，但是在性能上却不及后者，这也许表明了模型参数并非

决定模型性能的唯一因素，模型的架构、训练方法和数据的匹配度也有着重要的影响。

总的来说，这些结果揭示了在不同模型中选择最佳的模型进行任务需综合考虑多个因素，并且在知识库问答任务中，Galactica 和 LLaMA 模型可能是更优的选择。

G. 定性结果

根据加州大学伯克利分校的 Chatbot Arena 项目，我们大概了解了各个 LLM 的能力差距（如图 10）。不止本文提到的三个模型，在训练和测试的过程中也挑选了排行榜中的 ChatGLM、Vicuna 等开源项目，不过由于训练出错、效果较差，只能舍弃。



(a) 6 个代表性的 LLM 在 8 个类别中能力的比较

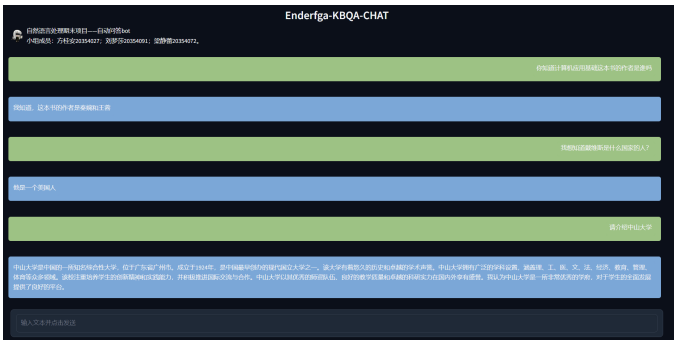
(b) 模型 A 在所有非平局的 A 对 B 战斗中获胜的比例。

Fig. 10. LLM 的能力对比

我们实验的初衷是实现基于 LLM 实现特定垂直领域的知识问答，故在我们的任务集之外，微调之后的 LLM 其实不一定能保证较好的泛化性能；同样即使是 Chatbot Arena 排行榜上最优的 GPT-4 也很难回复出我们任务集内的问题。为了验证这一猜想，我们搭建了一个 web 端的可交互自动问答 bot，来对比 LLaMA 与 GPT-4 的定性结果。

如图 11 所示，LLaMA 能够很好地回复任务集中的问题，即在垂直领域中完成了期望的效果，且还能维持一定的泛化性能，成功回复了“请介绍中山大学”这一任务集之外的问题。而 GPT-4 无法回复类似“计算机应用基础这本书的作者是谁”的自身知识外问题和“戴维斯是什么国家的人”的特定领域问题。

但是 GPT-4 优越的性能确保它能够进行合理分析，如“‘戴维斯’是一个常见的姓氏，特别是在英语讲的国家。最著名的可能是来自英国和美国。然而，由于它在世界各地都很常见，确切的国籍可能需要根据特定的人物背景来确定。”这一回复。当然 GPT-4 也不是万能的，其回复中也出现了错误，例如中山大学并没有五山校区，五山校区应该是华南理工大学的。



(a) LLaMA

(b) GPT-4

Fig. 11. LLM 的能力对比

V. 总结

在自然语言处理（NLP）领域，大型语言模型（Large Language Models，简称 LLM）如 OpenAI 的 GPT 系列模型和 Google 的 BERT 等，已成为当之无愧的主流，引领了一股以深度学习为核心，尤其是依赖于变换器架构（Transformer architecture）的研究潮流。

这些大型语言模型之所以备受赞誉，主要归功于其广泛的知识和理解能力。这些模型在大量的文本数据上进行预训练，从而获取和理解丰富的事实、观点和语言结构。然后，通过对具体任务进行微调，使其可以在各种 NLP 任务中展现卓越的表现，包括但不限于问答、文本分类、情感分析以及语义角色标注等。

然而，我们也必须正视大型语言模型的挑战。首先，

模型的训练需要消耗巨大的计算资源和数据，这对很多研究者和机构来说无疑是一道难以逾越的门槛。其次，由于这些模型可能会生成有偏见或不准确的输出，尤其是在训练数据本身就存在偏见或错误时，如何保证输出的公正性和准确性成为了另一个问题。此外，这些模型的内部结构和参数过于复杂，使得我们很难解释模型的决策过程。最后，我们还面临着如何防止模型被用于不道德或不合法的目的，比如生成假新闻或有害信息的风险。

对于特定垂直领域的知识问答任务，例如涉及医疗、法律或科技等领域，LLM 可以发挥至关重要的作用。通过从大量的未标记文本中学习语言模式和知识，LLM 具备了处理广泛的主题和问题类型的能力。这种灵活性在处理专业领域问题时表现出了显著优势，因为这些问题常常需要深厚的专业知识来解答。

通过微调技术，LLM 可以进一步适应特定的垂直领域。假如我们有一定数量的领域特定标记数据（例如问题-答案对），我们就可以进一步训练 LLM，让模型更好地理解和回答这个领域的问题。这种方式使得我们可以利用 LLM 创建一个专门针对特定领域问题的问答系统。

尽管如此，使用 LLM 处理垂直领域的知识问答任务仍然面临一些挑战。训练和微调模型需要大量的计算资源和领域特定的标记数据。此外，确保模型输出的准确性和可靠性是一个重要的挑战，尤其是在医疗和法律等领域，任何误解或错误的信息都可能带来严重的后果。

总的来说，虽然基于 LLM 实现特定垂直领域的知识问答是一种具有巨大潜力的方法，但在实践中需要进行精心的设计和谨慎的实施，以确保其实用性和安全性。