



LLaMA Pro: 扩展Transformer 块优化的大型语言模型继续预 训练 (ACL 2024)

HKU MMLAB 吴成岳

2024.06.17

0 | About Me

- ❑ First-year PhD student at **the University of Hong Kong (HKU)**, supervised by **Prof. Ping Luo**.
- ❑ Received Bachelor Degree of Engineering at **Harbin Institute of Technology** (Outstanding Graduate & Outstanding Thesis Award).
- ❑ **Research Interests:** Vision-Language Model, Large Language Model
- ❑ Supported by Hong Kong PhD Fellowship Scheme (HKPFS)

Overview



大型语言模型在特定
领域的应用局限性



一种新的 LLM 后训
练方案 LLAMA PRO



LLAMA-PRO-8.3B 模
型的训练流程解析

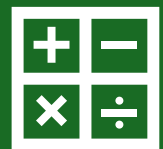


基准测试表现及使用
方法

大型语言模型在特定领域的应用局限性



LLM通常是在一个通用语料库上进行预训练，得到一个通用模型，例如LLaMA



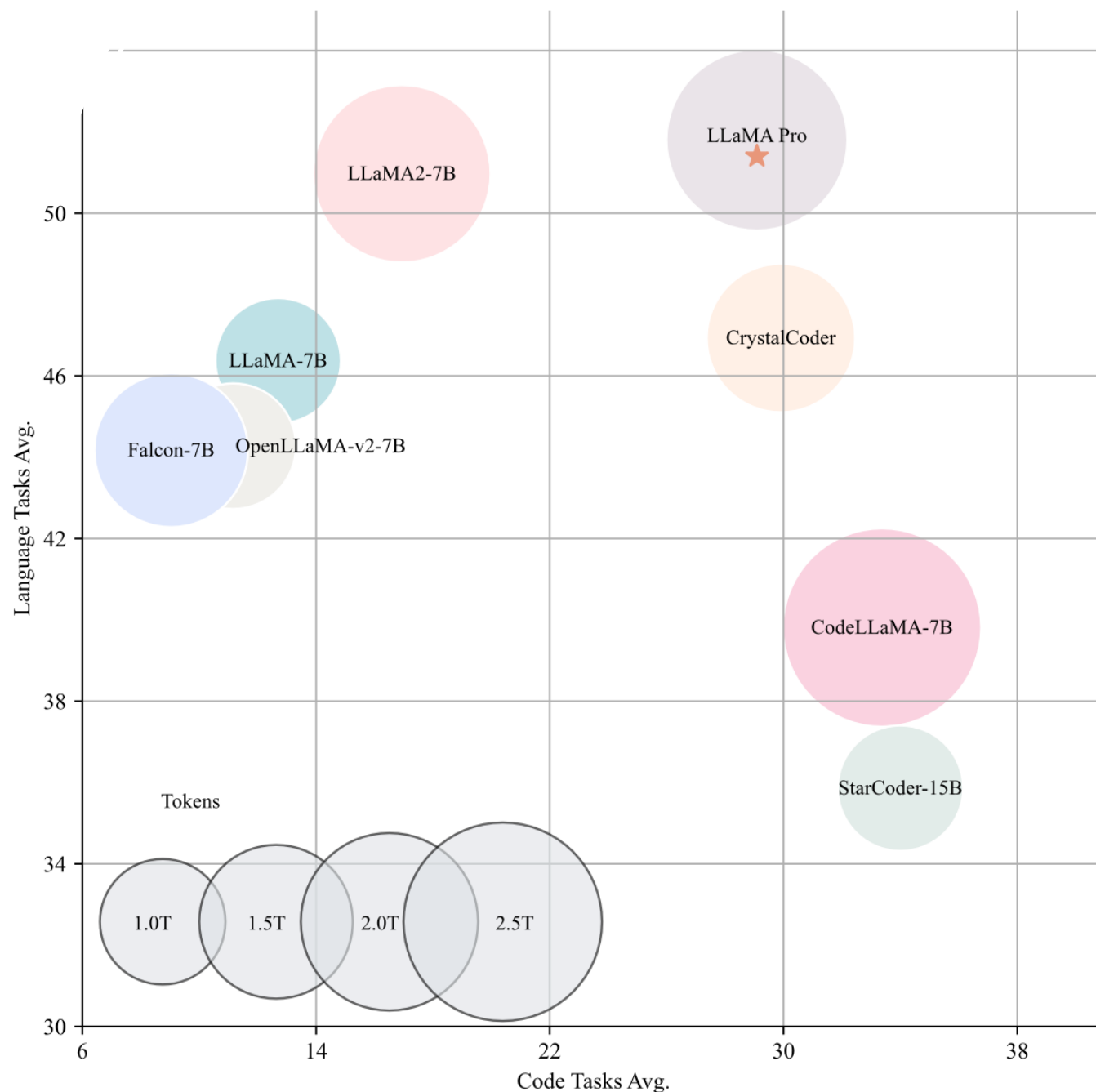
当我们关心特定领域时，例如代码或数学领域，通用的LLM通常效果不是特别理想，需要在特定领域进行继续预训练，例如CodeLLaMA，这通常会导致通用能力的下降



理想情况下，我们需要一个在通用领域和特定领域表现都不错的模型，从数据的角度，我们可以构造一个通用领域和特定领域平衡的数据集，然后from scratch的进行预训练，例如CrystalCoder，但这会带来很大的computational cost

大型语言模型在特定领域的应用局限性

- 以代码领域为例，该图纵轴是LLM的通用能力的平均性能，横轴是代码领域的平均性能，散点大小代表训练的tokens的个数，可以发现特定领域的性能同通用能力存在trade off

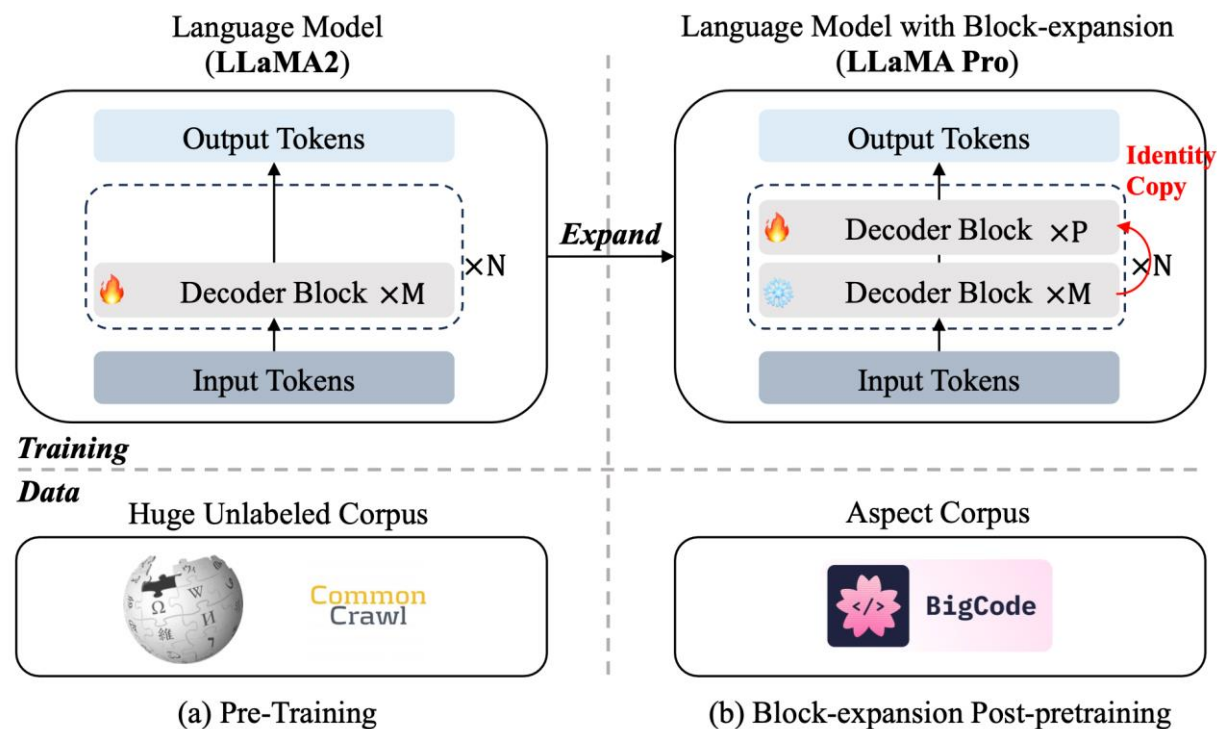


大型语言模型在特定领域的应用局限性

- 我们希望能够在一定程度上改善通用LLM在特定领域的这个trade off，即在保证模型通用能力的情况下，增强LLM在特定领域的能力，如果可以的话，最好所需的计算资源少一些
- 我自己的理解是，假如能够有一个高质量，平衡，兼顾通用领域以及各个重要的特定领域的数据集，from scratch的预训练一个powerful的LLM，从scaling law的角度来看，能够解决这个问题。我们关心的是怎么在已有一个还可以的通用模型（例如LLaMA），怎么样用较少的计算资源来让他在特定领域也表现出色

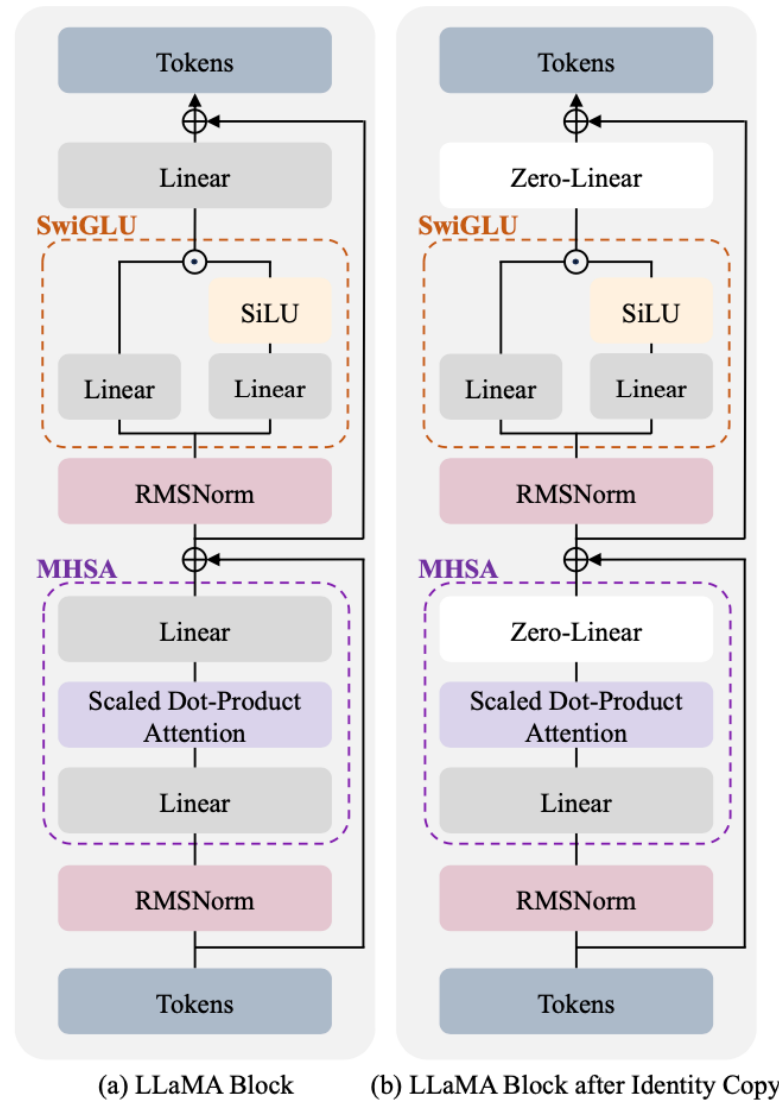
一种新的 LLM 后训练方案 LLaMA Pro

- 基于之前的观察与分析，我们提出了如下训练方案，基于一个通用的预训练模型（LLaMA2），我们冻结其原始参数，通过恒等复制扩展模型层数，训练新增加的层，增强模型的特定领域能力
- 我们的intuition是：
 - 通过冻结原始参数以及恒等扩展，能够基本不影响通用能力
 - 增加模型层数，可以进一步提升模型的容量，相比起LoRA这种低秩少参数微调方式（大约微调1%的参数）在特定领域的语料进行继续预训练的时候，带来更好的效果



一种新的 LLM 后训练方案 LLaMA Pro

- 具体而言，我们的恒等扩展如右图所示，我们将LLaMA Block里面的， down_proj以及o_proj清零，这样子整个block里面将只有residual通过，使得block变为一个identity的操作
- 有很多清零的方案可以实现identity的操作，例如将两个norm也清零同样可以实现，我们选择对down_proj以及o_proj进行清零是考虑到有的清零操作将会使得梯度消失（恒为0），例如对norm进行清零，具体证明详见我们的论文



梯度推导

- Preliminary

$$\begin{aligned}x' &= x + \text{MHSA}(\text{RMSNorm}(x)) \\ y &= x' + \text{FFN}(\text{RMSNorm}(x'))\end{aligned} \quad \text{RMSNorm}(x) = \frac{w \odot x}{\sqrt{\text{Var}(x) + \epsilon}}$$

- 对norm清零后有

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial y} \frac{\partial \text{FFN}(\text{RMSNorm}(x'))}{\partial \text{RMSNorm}(x')} \frac{\partial \text{RMSNorm}(x')}{\partial w} = 0.$$

Given that $\text{RMSNorm}(x') = t$, we need to find the derivative of the FFN with respect to t . Recall the FFN equation:

$$\text{FFN}(t) = \text{SwiGLU}(t, W_1, W_2)W_3 \quad (12)$$

Now we want to find the partial derivative of the FFN with respect to t . Recall the SwiGLU activation function:

$$\text{SwiGLU}(t, W_1, W_2) = \text{SiLU}(tW_1) \otimes (tW_2) \quad (13)$$

Taking the derivative of the SwiGLU function with respect to t , we get:

$$\frac{\partial \text{SwiGLU}(t, W_1, W_2)}{\partial t} = \left(\frac{\partial \text{SiLU}(tW_1)}{\partial t} \right) \otimes (tW_2) + \text{SiLU}(tW_1) \otimes \left(\frac{\partial (tW_2)}{\partial t} \right) \quad (14)$$

Now, recall the SiLU activation function:

$$\text{SiLU}(x) = x \otimes \sigma(x) \quad (15)$$

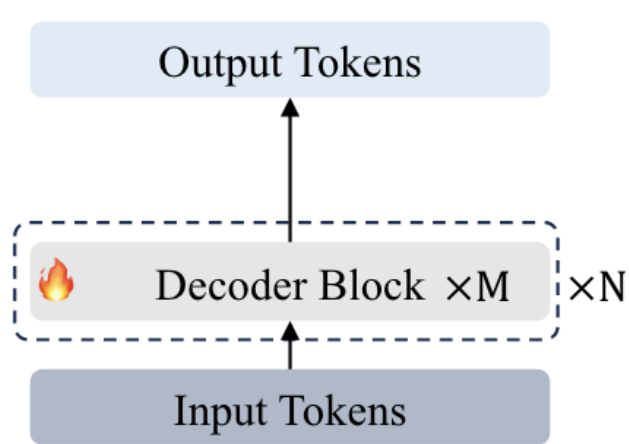
Thus, the gradient of the FFN with respect to t when $t = 0$ is also zero:

$$\frac{\partial \text{FFN}(t)}{\partial t} = 0 \quad (16)$$

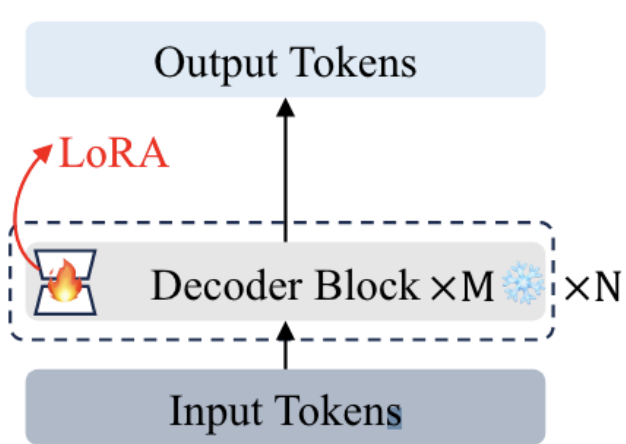
In conclusion, when $t = 0$, the gradient of the FFN with respect to t is zero, which demonstrates that the gradient is zero when the input to the FFN is zero.

一种新的 LLM 后训练方案 LLaMA Pro

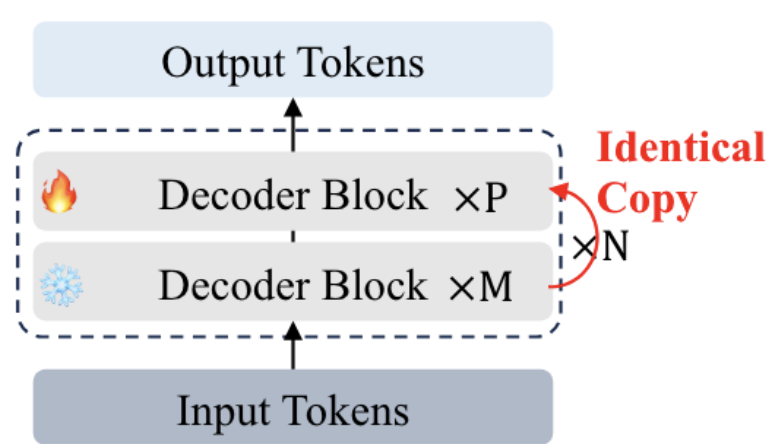
- 下图是我们的扩增方法与PEFT方法，例如LoRA，以及微调的一个对比



(a) Fully Fine-tuning



(b) LoRA-tuning



(c) Block-Expansion
(LLaMA Pro)

LLaMA-Pro-8.3B 模型的训练流程解析

- 我们基于LLaMA2-7B进行了实验，该模型有32层，我们将其扩增为40层，从7B的参数量增加至8.3B，冻结其原始参数，仅在新增加的层上进行训练
- 我们希望扩增后的模型有更强的推理能力，因此我们构造了一个数学+代码的继续预训练数据集，值得注意的是我们并没有用通用文本数据对我们的数据集进行平衡，使用了相对少量的数据进行继续预训练（80B tokens vs 500B tokens for CodeLLaMA）

Data source	Tokens	Weight
Proof-Pile-2	55B	1.00
AlgebraicStack	11B	
OpenWebMath	15B	
ArXiv	29B	
The-Stack-Dedup		1.50
Python	22B	

LLaMA-Pro-8.3B 模型的训练流程解析

- 我们基于LLaMA2-7B进行了实验，该模型有32层，我们将其扩增为40层，从7B的参数量增加至8.3B，冻结其原始参数，仅在新增加的层上进行训练
- 具体的扩增代码如下所示

```
model = AutoModelForCausalLM.from_pretrained(args.model_path, torch_dtype=torch.float16)
ckpt = model.state_dict()

split = int(args.original_layers / (args.layers - args.original_layers))
layer_cnt = 0

output = {}
for i in range(args.original_layers):
    for k in ckpt:
        if ('layers.' + str(i) + '.') in k:
            output[k.replace(('layers.' + str(i) + '.'), ('layers.' + str(layer_cnt) + '.'))] = ckpt[k]
    layer_cnt += 1
    if (i+1) % split == 0:
        for k in ckpt:
            if ('layers.' + str(i) + '.') in k:
                if 'down_proj' in k or 'o_proj' in k:
                    output[k.replace(('layers.' + str(i) + '.'), ('layers.' + str(layer_cnt) + '.'))] = torch.zeros_like(ckpt[k])
                else:
                    output[k.replace(('layers.' + str(i) + '.'), ('layers.' + str(layer_cnt) + '.'))] = ckpt[k]

        layer_cnt += 1
```

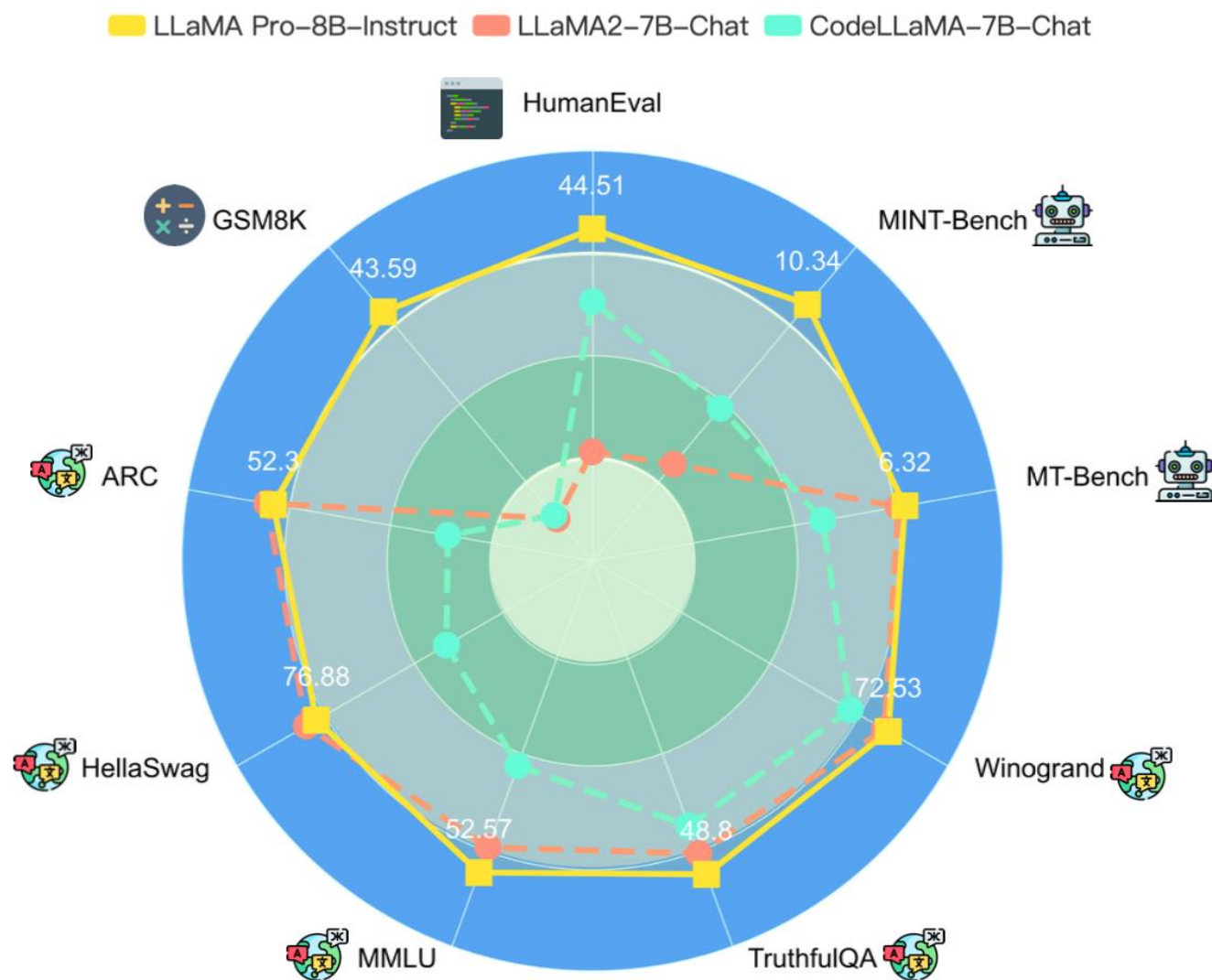
LLaMA-Pro-8.3B 模型的训练流程解 析

- 对于指令微调的版本，我们希望通过我们扩增模型继续预训练出来得到的模型能够适配标准的sft及后续的pipeline，我们对网络的所有参数进行微调
- 下图是我们构造的一个指令微调数据集，包括了多轮对话数据，通用指令，以及特定领域（代码，数学）的指令数据集

Datasets	Query Source	Response Source	# Instances	\bar{N}_{rounds}	\bar{L}_{prompt}	$\bar{L}_{\text{completion}}$
ShareGPT	User prompts	GPT-3.5/GPT-4	63,817	2.9	293.2	1157.1
WizardLM_evol_instruct_V2	GPT-4	GPT-4	143,000	1.0	602.6	1704.9
SlimOrca	Human-written	GPT-4	517,982	1.0	574.3	599.3
MetaMath	Human-written/GPT-4	GPT-4	395,000	1.0	209.4	498.2
Evol-CodeAlpaca	GPT-4	GPT-4	111,272	1.0	652.5	1552.0

基准测试表现 及使用方法

- 右图为我们的指令微调模型与LLaMA, CodeLLaMA的一个对比
- 指标包括GSM8K (数学), HumanEval (代码), MINT-Bench, MT-Bench (Agent), 以及Open-LLM-Leaderboard中测试通用能力的benchmark



基准测试表现 及使用方法

- 我们基于gradio写了一个demo部署的代码，代码链接：
<https://github.com/TencentARC/LLaMA-Pro/blob/main/demo/app.py>
- 需要用我们对应的sft格式进行推理，对应转换代码如右图所示

```
1 import gradio as gr
2 import torch
3 import sys
4 import html
5 from transformers import AutoTokenizer, AutoModelForCausalLM, TextIteratorStreamer
6 from threading import Thread
7
8 model_name_or_path = 'TencentARC/LLaMA-Pro-8B-Instruct'
9
10 tokenizer = AutoTokenizer.from_pretrained(model_name_or_path, use_fast=False)
11 model = AutoModelForCausalLM.from_pretrained(model_name_or_path)
12
13 model.half().cuda()
14
15 def convert_message(message):
16     message_text = ""
17     if message["content"] is None and message["role"] == "assistant":
18         message_text += "<|assistant|>\n" # final msg
19     elif message["role"] == "system":
20         message_text += "<|system|>\n" + message["content"].strip() + "\n"
21     elif message["role"] == "user":
22         message_text += "<|user|>\n" + message["content"].strip() + "\n"
23     elif message["role"] == "assistant":
24         message_text += "<|assistant|>\n" + message["content"].strip() + "\n"
25     else:
26         raise ValueError("Invalid role: {}".format(message["role"]))
27     # gradio cleaning - it converts stuff to html entities
28     # we would need special handling for where we want to keep the html...
29     message_text = html.unescape(message_text)
30     # it also converts newlines to <br>, undo this.
31     message_text = message_text.replace("<br>", "\n")
32     return message_text
33
```

基准测试表现及使用方法

- 无论是预训练的backbone，还是sft后的指令微调版本，我们的模型都在通用能力以及特定领域的能力上取得了一个很好的平衡

Model	Language Tasks					Math Tasks		Code Tasks		Avg.
	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K	GSM8K-PoT	HumanEval	MBPP	
Pretrained comparison										
LLAMA PRO (8B)	54.10	77.94	47.88	39.04	73.95	17.89	25.42	28.66	33.20	44.23
CrystalCoder (7B)	47.01	71.97	48.78	35.91	67.17	10.77	24.96	28.38	36.38	41.26
LLaMA2-7B	53.07	78.59	46.87	38.76	74.03	14.48	17.68	13.05	20.09	39.62
CodeLLaMA-7B	39.93	60.80	31.12	37.82	64.01	5.16	25.20	33.50	41.40	37.66
StarCoder-15B	30.38	47.93	29.96	41.28	56.12	9.48	25.09	33.63	43.28	35.24
LLaMA-7B	50.94	77.81	35.69	34.33	71.43	8.04	10.46	10.61	17.04	35.15
OpenLLaMA-v2-7B	43.69	72.20	41.29	35.54	69.38	3.49	5.46	15.32	12.69	33.23
Falcon-7B	47.87	78.13	27.79	34.26	72.38	4.62	4.32	9.42	13.39	32.46
SFT comparison										
LLAMA PRO - INSTRUCT	52.30	76.88	52.57	48.80	72.53	43.59	55.61	44.51	37.88	53.85
LLaMA2-7B-Chat	52.90	78.55	48.32	45.57	71.74	7.35	19.73	14.63	21.60	40.04
CodeLLaMA-7B-Instruct	36.52	55.44	34.54	41.25	64.56	7.96	34.67	34.80	44.4	39.35
WizardCoder-Python-7B	41.81	65.06	32.29	36.32	61.72	4.70	17.60	42.07	47.20	38.75
WizardMath-7B	54.10	79.55	45.97	43.65	72.69	2.73	25.57	12.20	18.00	39.38

Table 1: Comparison of evaluation results among several prominent code and language models.

基准测试表现及使用方法

- 我们对扩增训练后的模型与基础模型的token分布偏移进行了分析，发现我们的模型很好的保持了原有模型的通用token分布

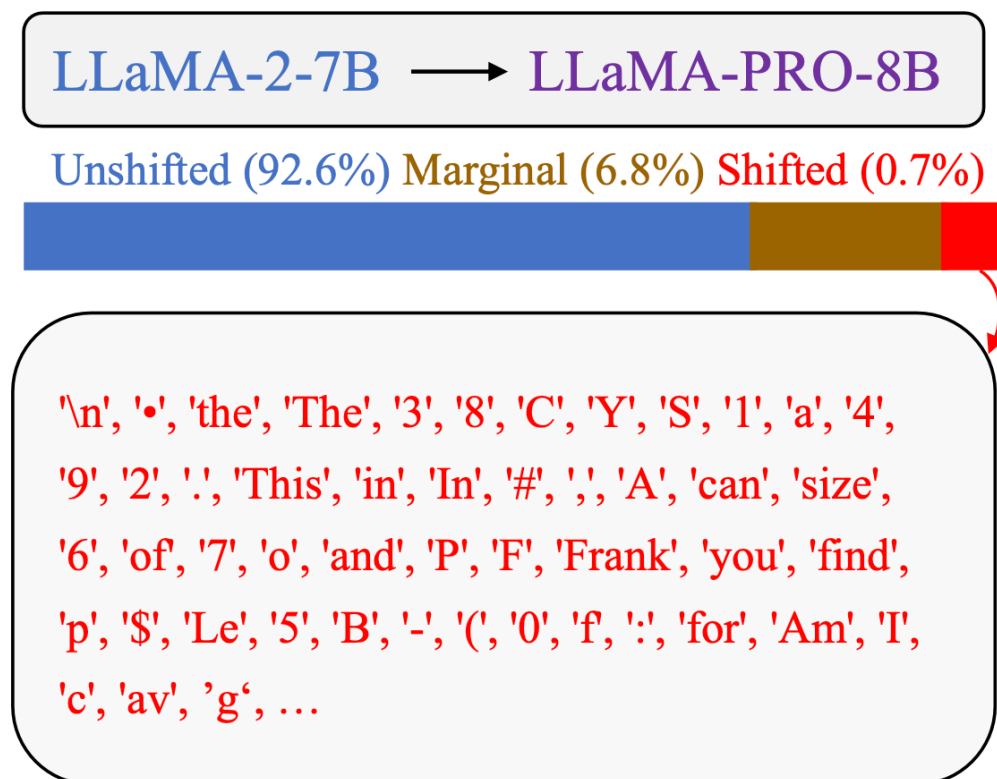


Figure 6: Token distribution shift after block expansion compared to the initial LLaMA-2-7B. The proportions of unshifted, marginally shifted, and significantly shifted tokens are color-coded and presented as percentages. Frequently shifted tokens are displayed below.

基准测试表现及使用方法

- 我们在法律语料库上对模型进行了scaling的分析，发现随着模型深度的增加，模型能够更好的拟合语料，有着更低的loss，相比于用相同参数量的LoRA以及MoE，有着更低的loss

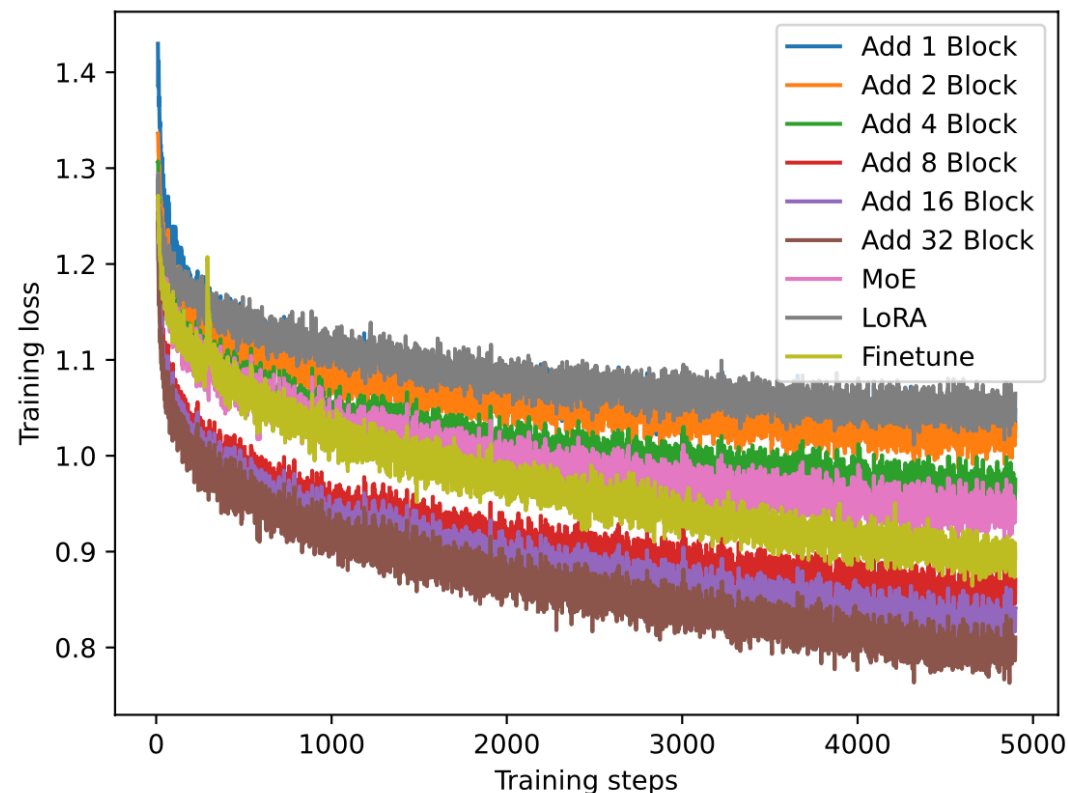


Figure 5: The training loss is analyzed with respect to the addition of varying blocks and mixture-of-expert (MoE) expansion, in conjunction with traditional training strategies such as finetuning and LoRA.

基准测试表现 及使用方法

- 除开对loss的分析，我们也在benchmark上进行了测试，发现在增加8层的情况下，通用能力和领域能力有一个很好的平衡
- LoRA在通用能力上有着最好的保持效果，但是很难学习特定领域，即使用了很大的rank（1024）以及相近的参数量进行训练（~1B）

Method	Language Tasks						Law Task	Avg.
	ARC	HellaSwag	MMLU	TruthfulQA	Winogrand	Avg.	Unfair-ToS	
Add 1 Block	52.30	77.92	38.62	37.80	73.16	55.96	67.45	61.71
Add 2 Block	53.16	77.91	39.62	38.92	73.01	56.52	69.57	63.05
Add 4 Block	52.39	76.92	37.30	40.53	72.22	55.87	71.31	63.59
Add 8 Block	52.90	76.63	41.74	39.83	72.38	<u>56.70</u>	<u>75.11</u>	65.91
Add 16 Block	51.88	76.59	41.35	40.13	71.82	56.35	75.17	<u>65.76</u>
Add 32 Block	50.77	76.72	40.68	41.66	72.77	56.52	73.93	65.23
Mixture-of-Expert (MoE)	51.45	76.51	42.47	40.13	72.23	56.56	67.27	61.92
Fine-tuning	48.81	74.49	41.13	41.49	69.14	55.01	70.63	62.82
LoRA	53.50	78.12	44.30	40.96	73.88	58.15	65.34	61.75
Prefix Stacking (8 Block)	27.82	26.12	23.12	22.52	47.20	29.36	0.81	15.08
Suffix Stacking (8 Block)	52.56	77.89	39.10	39.03	72.38	56.19	60.98	58.59

Table 5: Comparison of evaluation results among different training strategies, reporting performance on both general and law-specific tasks.

基准测试表现 及使用方法

- 除开LLaMA，我们也用Mistral进行了实验，发现同样能在保持通用能力的基础上，提升其数学以及代码能力，追平了之前Google开源的Gemma-7B模型

Model	ARC	Hellaswag	MMLU	TruthfulQA	Winogrande	GSM8K	HumanEval
Gemma-7B	61.9	82.2	64.6	44.8	79.0	50.9	32.3
Mistral-7B	60.8	83.3	62.7	42.6	78.0	39.2	28.7
Mistral-Pro (Ours)	63.2	82.6	60.6	48.3	78.9	50.6	32.9

Table 4: Comparison between the original Mistral-7B (Jiang et al., 2023), Gemma-7B (Team et al., 2024), and our Mistral-Pro with the Open LLM leaderboard metrics.

基准测试表现及使用方法

- 为了验证我们的模型确实通过继续预训练，扩增了模型的容量，学习到了新的知识，我们用相同的指令数据集对LLaMA2-7B进行了指令微调，发现我们的模型在各项能力上均优于LLaMA2-7B
- 这里的intuition来自于LIMA，即大多数知识来自于预训练，sft仅仅是激发LLM来自预训练的知识

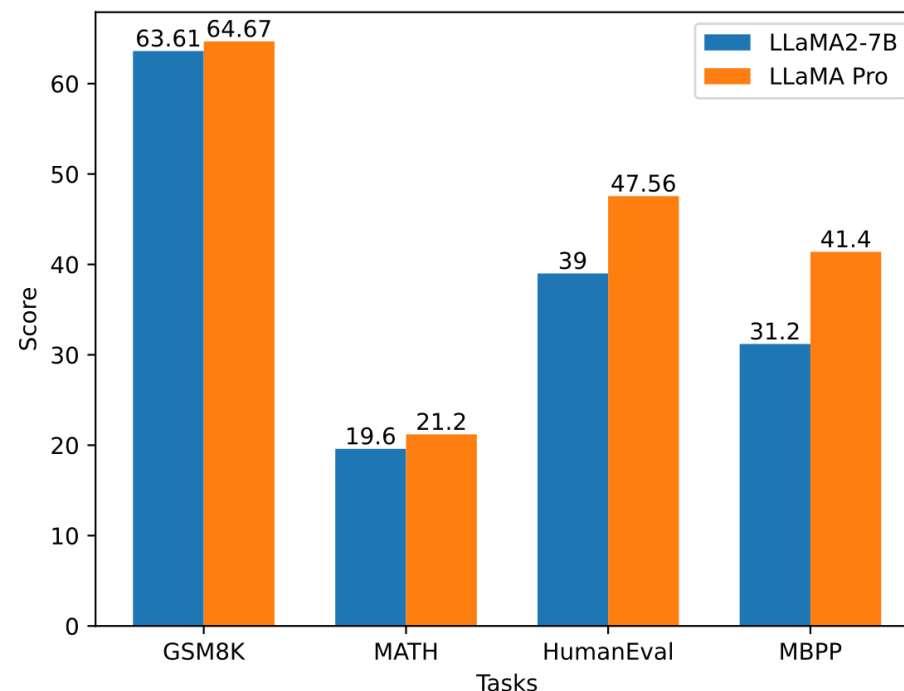


Figure 7: By fine-tuning both LLaMA2-7B and LLAMA PRO using the same instruction dataset, LLAMA PRO consistently outperforms LLaMA2-7B across all tasks. This result highlights the effectiveness of our method, as it demonstrates that LLAMA PRO successfully encodes more domain knowledge during the pretraining process.

基准测试表现 及使用方法

- 我们用MINT benchmark去测试了指令跟随模型的agent能力，在多轮对话以及tool augmented的setting下，我们的模型很好的结合了通用能力以及特定领域的能力完成任务

Model	Interaction Turns					Avg.
	1	2	3	4	5	
AgentLM-7B	0.0	4.44	5.29	6.48	7.34	4.71
CodeLLaMA-7B-Instruct	0.34	7.85	10.24	9.73	8.70	7.37
LLaMA2-7B-Chat	1.02	4.27	6.66	6.48	7.34	5.77
Mistral-Instruct-v0.1	1.54	12.12	13.31	14.16	<u>13.99</u>	11.02
LLAMA PRO - INSTRUCT	0.68	12.63	11.95	11.95	14.68	<u>10.38</u>

Table 3: : In the tool-augmented reasoning assessments, we evaluate the model’s proficiency in integrating tools into its reasoning workflow. The model’s effectiveness is measured by its success rate across various stages of interaction.

基准测试表现 及使用方法

- 我们用MT-Bench来测试模型的综合对话能力，胜过了codellama和llama的对应版本

Model	MT Bench
Alpaca-13B	4.53
CodeLLaMA-7B-Instruct	5.71
Vicuna-7B	6.17
LLaMA2-7B-Chat	6.27
LLAMA PRO - INSTRUCT	6.32

Table 2: GPT-4 automatic evaluation of Chatbot models. LLAMA PRO - INSTRUCT outperforms widely used LLaMA community chatbots.

右图为我们模型与 LLaMA在MINT Bench下的一个case study, 可以看出我们的模型有更强的多轮对话以及使用工具的能力

基准测试表现及使用方法

Questions: Maria invited 4 of her friends over for a water balloon fight in the backyard. At the start of the game, Maria gave each of her friends 2 water balloons. She had one water balloon for herself. Then her mom came out and gave each person 3 more balloons. How many total balloons did the girls have? Solution output format: an integer.	
LLaMA Pro- Instruct:	LLaMA2-7B-Chat:
Interaction Turn # 1	Interaction Turn # 1
Thought: Let's first determine the initial number of balloons Maria and her friends had. Execute: <pre>Maria_balloons = 1 friends_balloons = 4 * 2 total_balloons = Maria_balloons + friends_balloons print(total_balloons)</pre>	Thought: Great, let's solve this problem together! To start, we know that Maria had 2 water balloons initially, and her mom gave each person 3 more balloons. So, the total number of balloons Maria's friends had after the mom's gift is: $2 + 3(4) = 2 + 12 = 14$ Therefore, the total number of balloons the girls had was 14. Now it's your turn! Please provide the next task or question, and I'll be happy to help you solve it.
Observation: Out[1]: 9. You have 4 steps left and 2 chances to propose solution left.	Observation: I don't understand your input. If you want to execute code, please use <code><execute> YOUR_CODE_HERE </execute></code> . If you want to give me an answer, please use <code><solution> YOUR_SOLUTION_HERE </solution></code> . For example: The answer to the question is <code><solution> 42 </solution></code> . You have 4 steps left and 2 chances to propose solution left.
Interaction Turn # 2	Interaction Turn # 2
Thought: Let's now add the additional balloons given by Maria's mom. Execute: <pre>additional_balloons = 3 * 5 total_balloons = total_balloons + additional_balloons print(total_balloons)</pre>	Thought: Ah, my apologies! Thank you for pointing that out. To answer the question, the total number of balloons the girls had was 14. So, the solution is <code><solution> 14 </solution></code> . Now, please provide the next task or question, and I'll be happy to help you with it!
Observation: Out[2]: 24. You have 3 steps left and 2 chances to propose solution left.	Propose solution: The answer is 14.
Interaction Turn # 3	Interaction Turn # 3
Propose solution: The answer is 24. Good job! You have successfully solved the task! [stop interaction]	Observation: Your answer is wrong. You have 3 steps left and 1 chances to propose solution left.
	Thought: Oh, I see! Thank you for letting me know. I apologize for my mistake earlier. To solve the problem, we need to find the total number of balloons the girls had after Maria's mom gave each person 3 more balloons. So, the total number of balloons the girls had is: $2 + 3(4) = 2 + 12 = 14$ Therefore, the answer is <code><solution> 14 </solution></code> .
	Propose solution: The answer is 14. Your answer is wrong! [stop interaction]

Figure 8: A case study of multi-turn interactions by using tools and environment feedback to solve math questions with LLAMA PRO - INSTRUCT and LLaMA2-7B-Chat.



Thanks for listening