The Llama 3 Herd of Models

Llama Team https://llama.meta.com/

AI @ Meta

Editor: A detailed contributor list can be found in the appendix of this paper.

Abstract

现代人工智能（AI）系统由基础模型驱动。本文介绍了一套新的基础模型，称为 Llama 3。它是一群本地支持多语言、编码、推理和工具使用的语言模型。我们最大的模型是一个具有 4050 亿参数和高达 128K 令牌上下文窗口的密集 Transformer。本文对 Llama 3 进行了广泛的实证评估。我们发现 Llama 3 在众多任务上提供了与 GPT-4 等领先语言模型相当的质量。我们公开发布了 Llama 3，包括 4050 亿参数语言模型的预训练和后训练版本，以及我们的 Llama Guard 3 模型，用于输入和输出安全。本文还介绍了我们通过组合方法将图像、视频和语音能力集成到 Llama 3 中的实验结果。我们观察到这种方法在图像、视频和语音识别任务上与最先进的技术竞争。生成的模型尚未广泛发布，因为它们仍在开发中。

Keywords: keyword one, keyword two, keyword three

1 Introduction

基础模型是为语言、视觉、语音和/或其他模态设计的通用模型，旨在支持大量 AI 任务。它们构成了许多现代 AI 系统的基础。

现代基础模型的发展包括两个主要阶段：(1) 预训练阶段，在这个阶段，模型使用简单的任务（如下一个词预测或字幕生成）进行大规模训练；(2) 后训练阶段，在这个阶段，模型被调整以遵循指令、与人类偏好对齐，并提高特定能力（例如，编码和推理）。

在本文中，我们介绍了一套新的语言基础模型，称为 Llama 3。Llama 3 模型群原生的支持多语言、编码、推理和工具使用。我们最大的模型是一个具有 4050 亿参数的密集 Transformer，能够在高达 128K 令牌的上下文中处理信息。表 1 列出了每个模型成员。本文中呈现的所有结果都是针对 Llama 3.1 模型的，为了简洁，我们将在全文中称之为 Llama 3。

我们相信，在开发高质量的基础模型方面有三个关键的杠杆：数据、规模和管理复杂性。我们在开发过程中寻求优化这三个杠杆：

- 数据。与 Llama 的早期版本（Touvron 等人，2023a，b）相比，我们改进了用于预训练和后训练的数据量和质量。这些改进包括开发更谨慎的预训练数据处理和策划流程，以及为后训练数据开发更严格的质量保证和过滤方法。我们对 Llama 3 进行了大约 15T 多语言令牌的预训练，而 Llama 2 是 1.8T 令牌。

- 规模。我们训练的模型规模远远大于以前的 Llama 模型：我们的旗舰语言模型使用了 3.8×10^{25} 次浮点运算进行预训练，比 Llama 2 的最大版本多近 50 倍。具体来说，我们在 15.6T 文本令牌上预训练了一个具有 4050 亿可训练参数的旗舰模型。根据基础模型的规模法则，我们的旗舰模型的表

	Finetuned	Multilingual	Long context	Tool use	Release
Llama 3 8B	✗	✗[1]	✗	✗	April 2024
Llama 3 8B Instruct	✓	✗	✗	✗	April 2024
Llama 3 70B	✗	✗[1]	✗	✗	April 2024
Llama 3 70B Instruct	✓	✗	✗	✗	April 2024
Llama 3.1 8B	✗	✓	✓	✗	July 2024
Llama 3.1 8B Instruct	✓	✓	✓	✓	July 2024
Llama 3.1 70B	✗	✓	✓	✗	July 2024
Llama 3.1 70B Instruct	✓	✓	✓	✓	July 2024
Llama 3.1 405B	✗	✓	✓	✗	July 2024
Llama 3.1 405B Instruct	✓	✓	✓	✓	July 2024

图 1: 表 1 Llama 3 模型群的概览。本文中的所有结果都是针对 Llama 3.1 模型的。

现超过了使用相同过程训练的较小模型。虽然我们的规模法则表明，我们的旗舰模型对于我们的训练预算来说是近似计算最优的大小，我们还训练了更小的模型，比计算最优的时间长得多。产生的模型在相同的推理预算下比计算最优模型表现更好。我们使用旗舰模型在后训练期间进一步提高这些较小模型的质量。

- 管理复杂性。我们做出设计选择，力求最大化我们扩展模型开发过程的能力。例如，我们选择标准的密集 Transformer 模型架构（Vaswani 等人，2017）进行小幅度调整，而不是选择专家混合模型（Shazeer 等人，2017）以最大化训练稳定性。同样，我们采用了相对简单的后训练程序，基于监督微调（SFT）、拒绝采样（RS）和直接偏好优化（DPO；Rafailov 等人（2023）），而不是更复杂的强化学习算法（Ouyang 等人，2022；Schulman 等人，2017），这些算法往往不太稳定，更难扩展。

我们的工作成果是 Llama 3：一个包含 8B、70B 和 405B 参数的三种多语言的语言模型的群体。我们在涵盖广泛语言理解任务的众多基准数据集上评估了 Llama 3 的性能。此外，我们进行了广泛的人类评估，将 Llama 3 与竞争模型进行了比较。表 2 展示了旗舰 Llama 3 模型在关键基准测试中的表现概览。我们的实验评估表明，我们的旗舰模型在各种任务上的表现与 GPT-4（OpenAI，2023a）等领先的语言模型相当，并且接近于达到最先进的水平。我们的较小模型是同类中最好的，超过了参数数量相似的替代模型（Bai 等人，2023；Jiang 等人，2023）。Llama 3 还在有益性和无害性之间提供了比其前身（Touvron 等人，2023b）更好的平衡。我们在第 5.4 节中对 Llama 3 的安全性进行了详细分析。

我们正在根据 Llama 3 社区许可证的更新版本公开发布所有三个 Llama 3 模型；请参见 `https://llama.meta.com` 这包括我们的 405B 参数语言模型的预训练和后训练版本，以及我们的 Llama Guard 模型（Inan 等人，2023）的新版本，用于输入和输出安全。我们希望旗舰模型的开放发布将激发研究社区的一波创新，并加速朝着负责任的人工通用智能（AGI）发展道路前进。

作为 Llama 3 开发过程的一部分，我们还开发了模型的多模态扩展，使其具备图像识别、视频识别和语音理解能力。这些模型仍在积极开发中，尚未准备好发布。除了我们的语言建模结果外，本文还展示了我们对这些多模态模型进行的初步实验的结果。

2 General Overview

Llama 3 的模型架构在图 1 中进行了说明。我们 Llama 3 语言模型的开发包括两个主要阶段：

Category	Benchmark	Llama 3 8B	Gemma 2 9B	Mistral 7B	Llama 3 70B	Mixtral 8x22B	GPT 3.5 Turbo	Llama 3 405B	Nemotron 4 340B	GPT-4 (0125)	GPT-4o	Claude 3.5 Sonnet
General	MMLU (5-shot)	69.4	**72.3**	61.1	**83.6**	76.9	70.7	87.3	82.6	85.1	89.1	**89.9**
	MMLU (0-shot, CoT)	**73.0**	72.3△	60.5	**86.0**	79.9	69.8	88.6	78.7◁	85.4	**88.7**	88.3
	MMLU-Pro (5-shot, CoT)	**48.3**	–	36.9	**66.4**	56.3	49.2	73.3	62.7	64.8	74.0	**77.0**
	IFEval	**80.4**	73.6	57.6	**87.5**	72.7	69.9	**88.6**	85.1	84.3	85.6	88.0
Code	HumanEval (0-shot)	**72.6**	54.3	40.2	**80.5**	75.6	68.0	89.0	73.2	86.6	90.2	**92.0**
	MBPP EvalPlus (0-shot)	**72.8**	71.7	49.5	**86.0**	78.6	82.0	88.6	72.8	83.6	87.8	**90.5**
Math	GSM8K (8-shot, CoT)	**84.5**	76.7	53.2	**95.1**	88.2	81.6	**96.8**	92.3◇	94.2	96.1	96.4◇
	MATH (0-shot, CoT)	**51.9**	44.3	13.0	**68.0**	54.1	43.1	73.8	41.1	64.5	**76.6**	71.1
Reasoning	ARC Challenge (0-shot)	83.4	**87.6**	74.2	**94.8**	88.7	83.7	**96.9**	94.6	96.4	96.7	96.7
	GPQA (0-shot, CoT)	32.8	–	28.8	**46.7**	33.3	30.8	51.1	–	41.4	53.6	**59.4**
Tool use	BFCL	**76.1**	–	60.4	84.8	–	**85.9**	88.5	86.5	88.3	80.5	**90.2**
	Nexus	**38.5**	30.0	24.7	**56.7**	48.5	37.2	**58.7**	–	50.3	56.1	45.7
Long context	ZeroSCROLLS/QuALITY	81.0	–	–	90.5	–	–	**95.2**	–	**95.2**	90.5	90.5
	InfiniteBench/En.MC	65.1	–	–	78.2	–	–	**83.4**	–	72.1	82.5	–
	NIH/Multi-needle	98.8	–	–	97.5	–	–	98.1	–	**100.0**	**100.0**	90.8
Multilingual	MGSM (0-shot, CoT)	**68.9**	53.2	29.9	**86.9**	71.1	51.4	**91.6**	–	85.9	90.5	**91.6**

图 2: 表 2 在关键基准评估中微调后的 Llama 3 模型的性能。该表比较了 8B、70B 和 405B 版本的 Llama 3 与竞争模型的性能。我们在三种模型尺寸等级中，每个等级中表现最佳模型的结果用粗体表示。 表示使用 5 次提示（无 CoT）获得的结果。 表示未使用 CoT 获得的结果。 表示使用零次提示获得的结果。

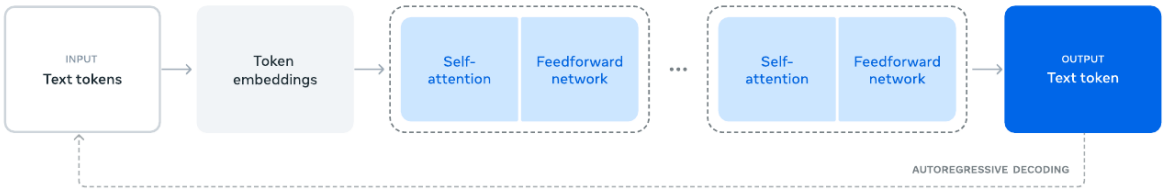

图 3: 图 1 Llama 3 的整体架构和训练示意图。Llama 3 是一个 Transformer 语言模型，被训练用于预测文本序列的下一个标记。详见正文。

- 语言模型预训练。我们首先将一个大型多语言文本语料库转换为离散的标记，并在生成的数据上预训练一个大型语言模型（LLM）以执行下一个标记预测。在语言模型预训练阶段，模型学习语言的结构，并从它“阅读”的文本中获得大量关于世界的知识。为了有效地做到这一点，预训练是在大规模上进行的：我们在 8K 标记的上下文中，使用 15.6T 标记预训练了一个具有 4050 亿参数的模型。这个标准的预训练阶段之后是一个持续预训练阶段，它将支持的上下文窗口增加到 128K 标记。见第 3 节了解详细信息。

- 语言模型后训练。预训练的语言模型对语言有丰富的理解，但尚未遵循指令或以我们期望助手表现的方式行事。我们通过几轮人类反馈来调整模型，每轮都涉及在指令调整数据上进行监督微调（SFT）和直接偏好优化（DPO；Rafailov 等人，2024）。在这个后训练阶段，我们还整合了新的能力，例如工具使用，并观察到在其他领域，如编码和推理方面的显著改进。见第 4 节了解详细信息。最后，安全缓解措施也在后训练阶段整合到模型中，具体细节在第 5.4 节中描述。

生成的模型具有丰富的功能集。它们可以用至少八种语言回答问题，编写高质量的代码，解决复杂的推理问题，并开箱即用或以零样本方式使用工具。我们还进行实验，通过组合方法将图像、视频和语音能力添加到 Llama 3 中。我们研究的方法包括图 28 所示的三个附加阶段：

- 多模态编码器预训练。我们为图像和语音训练单独的编码器。我们在大量图像-文本对上训练我们的图像编码器。这教会了模型视觉内容与自然语言中对该内容描述之间的关系。我们的语音编码器是使用一种自监督方法训练的，该方法遮蔽输入语音的部分并尝试通过离散标记表示重构被遮蔽的部分。结果，模型学习了语音信号的结构。见第 7 节了解图像编码器的详细信息和第 8 节了解语音编码器的详细信息。
- 视觉适配器训练。我们训练一个适配器，将预训练的图像编码器集成到预训练的语言模型中。适配器由一系列交叉注意力层组成，将图像编码器的表示输入到语言模型中。适配器在文本-图像对上进行训练。这将图像表示与语言表示对齐。在适配器训练期间，我们还更新了图像编码器的参数，但我们故意不更新语言模型的参数。我们还基于图像适配器在配对视频-文本数据上训练了一个视频适配器。这使模型能够跨帧聚合信息。见第 7 节了解详细信息。
- 语音适配器训练。最后，我们通过一个适配器将语音编码器集成到模型中，该适配器将语音编码转换为可以直接输入到微调后的语言模型中的标记表示。在监督微调阶段，适配器和编码器的参数被联合更新，以实现高质量的语音理解。在语音适配器训练期间，我们不更改语言模型。我们还集成了一个文本到语音系统。见第 8 节了解详细信息。

我们的多模态实验导致模型能够识别图像和视频的内容，并通过语音接口支持交互。这些模型仍在开发中，尚未准备好发布。

3 Pre-Training

语言模型预训练包括：(1) 大规模训练语料库的策划和过滤，(2) 开发模型架构和相应的规模法则以确定模型大小，(3) 开发大规模高效预训练技术，以及 (4) 开发预训练配方。我们在下面分别介绍这些组成部分。

3.1 预训练数据

我们从包含直到 2023 年底的知识的多种数据源中创建语言模型预训练数据集。我们对每个数据源应用了几种去重方法和数据清洗机制，以获得高质量的标记。我们移除了包含大量个人可识别信息 (PII) 的域名 (Domain)，以及已知含有成人内容的域名。

3.1.1 网络数据策划 (Curation)

我们使用的大部分数据来自网络，我们下面描述我们的清洗过程。

PII 和安全过滤 在其他缓解措施中，我们实施了旨在移除可能包含不安全内容或大量 PII 数据的网站数据的过滤器，根据各种 Meta 安全标准被评为有害的域名，以及已知含有成人内容的域名。

文本提取和清洗 我们处理非截断网页文档的原始 HTML 内容，提取高质量多样化文本。为此，我们构建了一个自定义解析器，提取 HTML 内容，并优化了在去除通用模板和内容召回方面的精确度。我们通过人类评估来评估我们解析器的质量，将其与优化文章类内容的流行的第三方 HTML 解析器进行比较，并发现它表现良好。我们仔细处理包含数学和代码内容的 HTML 页面，以保留该内容的结构。我们保留图像 alt 属性文本，因为数学内容通常表示为预先渲染的图像，其中 alt 属性也提供了数学内容。我们实验性地评估不同的清洗配置。我们发现，与纯文本相比，Markdown 对主要在网络数据上训练的模型的性能有害，因此我们移除了所有 Markdown 标记。

去重 我们在 URL、文档和行级别应用了几轮去重：

- **URL 级别去重**我们对整个数据集进行 URL 级别去重。我们保留每个 URL 对应页面的最新版本。
- **文档级别去重**我们对整个数据集进行全局 MinHash 去重，以移除几乎重复的文档。
- **行级别去重**我们执行与 ccNet 类似的积极行级别去重。我们移除在每 30M 文档的桶中出现超过 6 次的行。

尽管我们的手动定性分析显示，行级别去重不仅移除了来自各种网站（如导航菜单、Cookie 警告）的剩余通用模板，还移除了频繁出现的高质量文本，但我们的实证评估显示了强有力的改进。

启发式过滤 我们开发了启发式规则，以移除额外的低质量文档、异常值和重复过多的文档。一些启发式的例子包括：

- 我们使用重复 n-gram 覆盖率（Rae 等人，2021）来移除由重复内容（如日志记录或错误消息）组成的行。这些行可能非常长且唯一，因此无法通过行去重过滤。
- 我们使用“脏字”计数（Raffel 等人，2020）来过滤出不在域名阻止列表覆盖范围内的成人网站。
- 我们使用令牌分布 Kullback-Leibler 散度来过滤出与训练语料库分布相比含有过多异常令牌的文档。

基于模型的质量过滤 此外，我们尝试应用各种基于模型的质量分类器来选择高质量的标记。这些包括使用 fasttext（Joulin 等人，2017）这样的快速分类器，训练以识别给定文本是否会被维基百科引用（Touvron 等人，2023a），以及更计算密集的基于 Roberta 的分类器（Liu 等人，2019a）在 Llama 2 预测上训练。为了基于 Llama 2 训练一个质量分类器，我们创建了一个清洗过的网络文档的训练集，描述了质量要求，并指示 Llama 2 的聊天模型确定文档是否满足这些要求。我们使用 DistilRoberta（Sanh 等人，2019）为每个文档生成质量分数，以提高效率。我们通过实验评估了各种质量过滤配置的有效性。

代码和推理数据 与 DeepSeek-AI 等人（2024）类似，我们构建了特定领域的管道，提取代码和与数学相关的网页。具体来说，代码和推理分类器都是基于 Llama 2 在网络数据上进行训练的 DistilledRoberta 模型。与上述一般质量分类器不同，我们进行了提示调整，以针对包含数学演绎、STEM 领域的推理和与自然语言交织的代码的网页。由于代码和数学的令牌分布与自然语言的分布大不相同，这些管道实现了特定领域的 HTML 提取、定制的文本特征和过滤启发式规则。

多语言数据 与我们上面描述的英文处理管道类似，我们实施了过滤器，以移除可能包含 PII 或不安全内容的网站的数据。我们的多语言文本处理管道有几个独特的特点：

- 我们使用基于 fasttext 的语言识别模型将文档分类为 176 种语言。
- 我们在每种语言的数据中执行文档级别和行级别的去重。
- 我们应用特定语言的启发式和基于模型的过滤器来移除低质量文档。

此外，我们使用基于多语言 Llama 2 的分类器对多语言文档进行质量排名，以确保优先考虑高质量内容。我们通过实验确定预训练中使用的多语言标记的数量，平衡模型在英文和多语言基准上的性能。

3.1.2 确定数据混合

要获得高质量的语言模型，仔细确定预训练数据混合中不同数据源的比例至关重要。我们在确定数据混合的主要工具是知识分类和规模法则实验。

知识分类 我们开发了一个分类器，对我们的网络数据包含的信息类型进行分类，以更有效地确定数据混合。我们使用这个分类器来降低在网络上过度表示的数据类别，例如艺术和娱乐。

数据混合的规模法则 为确定最佳数据混合，我们执行规模法则实验，其中我们训练几个小型模型在数据混合上，并使用该数据混合预测大型模型在该混合上的性能（见第 3.2.1 节）。我们多次重复这个过程，以选择不同的数据混合，然后在这个候选数据混合上训练一个更大的模型，并评估该模型在几个关键基准上的性能。

数据混合摘要 我们最终的数据混合大致包含 50% 的对应于一般知识标记，25% 的数学和推理标记，17% 的代码标记，以及 8% 的多语言标记。

3.1.3 数据退火处理

通过实证研究，我们发现在小量的高质量代码和数学数据上进行退火处理（见 3.4.3 节）可以提升预训练模型在关键基准测试上的性能。与 Li 等人（2024b）类似，我们使用一种数据混合进行退火处理，该混合在选定的领域中上采样了高质量数据。在我们的退火数据中，我们不包括常用基准测试中的任何训练集。这使我们能够评估 Llama 3 的真实少次学习能力和跨领域泛化能力。

遵循 OpenAI（2023a），我们评估了在 GSM8k（Cobbe 等人，2021）和 MATH（Hendrycks 等人，2021b）训练集上进行退火处理的有效性。我们发现，退火处理改善了预训练的 Llama 3 8B 模型在 GSM8k 和 MATH 验证集上的性能，分别提高了 24.0% 和 6.4%。然而，在 405B 模型上的性能提升可以忽略不计，这表明我们的旗舰模型具有强大的上下文学习和推理能力，不需要特定的领域内训练样本就能获得出色的性能。

利用退火评估数据质量 与 Blakeney 等人（2024）类似，我们发现退火使我们能够判断小规模特定领域数据集的价值。我们通过将一个训练了 50% 的 Llama 3 8B 模型的学习率在线性地退火到 0 的 40B 个令牌上，来衡量这些数据集的价值。在这些实验中，我们给新数据集分配了 30% 的权重，其余 70% 的权重分配给了默认数据混合。使用退火来评估新数据源比为每个小数据集执行规模法则实验更有效率。

3.2 模型架构

Llama 3 使用标准的密集 Transformer 架构（Vaswani 等人，2017）。在模型架构方面，它与 Llama 和 Llama 2（Touvron 等人，2023a,b）相比没有显著偏离；我们的性能提升主要是由于数据质量和多样性的提高以及训练规模的增加。

与 Llama 3 相比，我们做了一些小的修改：

- 我们使用分组查询注意力（GQA；Ainslie 等人，2023）和 8 个键值头来提高推理速度，并在解码过程中减少键值缓存的大小。
- 我们使用注意力掩码来防止同一序列内不同文档之间的自注意力。我们发现这一变化在标准预训练中的影响有限，但在对非常长序列的持续预训练中很重要。
- 我们使用一个 128K 令牌的词汇表。我们的令牌词汇表结合了来自 tiktoken3 分词器的 100K 令牌和额外的 28K 令牌，以更好地支持非英语语言。与 Llama 2 分词器相比，我们的新分词器将样本英语数据的压缩率从 3.17 提高到 3.94 个字符每令牌。这使模型能够在相同的训练计算量下 “阅读” 更多的文本。我们还发现，添加来自选定非英语语言的 28K 令牌既提高了压缩比率，也提高了下游性能，而对英语分词没有影响。
- 我们将 RoPE 基础频率超参数增加到 500,000。这使我们能够更好地支持更长的上下文；Xiong 等人（2023）表明，这个值对于长达 32,768 的上下文是有效的。

Llama 3 405B 使用了一个具有 126 层、16,384 个令牌表示维度和 128 个注意力头的架构；详见表 3。这导致模型大小根据我们数据上的规模法则，对于我们的 3.8×10^{25} FLOPs 训练预算来说，是大约计算最优的。

	8B	70B	405B
Layers	32	80	126
Model Dimension	4,096	8192	16,384
FFN Dimension	6,144	12,288	20,480
Attention Heads	32	64	128
Key/Value Heads	8	8	8
Peak Learning Rate	3×10^{-4}	1.5×10^{-4}	8×10^{-5}
Activation Function	SwiGLU		
Vocabulary Size	128,000		
Positional Embeddings	RoPE ($\theta = 500,000$)		

图 4: 表 3 Llama 3 的关键超参数概览。我们为 8B、70B 和 405B 语言模型显示了设置。

3.2.1 规模法则

我们发展规模法则（Hoffmann 等人，2022；Kaplan 等人，2020）来确定我们的旗舰模型的最优大小，考虑到我们的预训练计算预算。除了确定最优模型大小之外，一个主要挑战是预测旗舰模型在下游基准任务上的性能，由于几个问题：(1) 现有的规模法则通常只预测下一个标记预测损失，而不是特定的

基准性能。(2) 规模法则可能嘈杂且不可靠，因为它们是基于使用小计算预算进行的预训练运行来发展的（Wei 等人，2022b）。

为了应对这些挑战，我们实施了一个两阶段方法来发展能够准确预测下游基准性能的规模法则：

1. 我们首先建立计算最优模型在下游任务上的负对数似然与训练 FLOPs 之间的相关性。
2. 接下来，我们利用规模法则模型和使用更高计算 FLOPs 训练的旧模型，将下游任务上的负对数似然与任务准确率相关联。在这一步中，我们特别利用了 Llama 2 系列模型。

这种方法使我们能够在给定特定数量的训练 FLOPs 的情况下，为计算最优模型预测下游任务的性能。我们使用类似的方法选择我们的预训练数据混合（见第 3.4 节）。

规模法则实验 具体来说，我们通过在 6×10^{18} FLOPs 到 10^{22} FLOPs 之间的计算预算下预训练模型来构建我们的规模法则。在每个计算预算下，我们预训练大小在 4000 万到 160 亿参数之间的模型，每个计算预算下使用子集的模型大小。在这些训练运行中，我们使用余弦学习率计划，并在 2000 个训练步骤中进行线性预热。峰值学习率根据模型大小设置在 2×10^{-4} 到 4×10^{-4} 之间。我们将余弦衰减设置为峰值的 0.1。每个步骤的权重衰减设置为该步骤学习率的 0.1 倍。我们为每个计算规模使用固定的批量大小，范围在 25 万到 400 万之间。

这些实验产生了图 2 中的 IsoFLOPs 曲线。这些曲线上的损失是在单独的验证集上测量的。我们使用二度多项式拟合测量的损失值，并确定每个抛物线的最小值。我们将抛物线的最小值称为相应预训练计算预算的计算最优模型。

我们用这种方式识别出的计算最优模型来预测特定计算预算下的最佳训练令牌数量。为此，我们假设计算预算 C 和最佳训练令牌数量 $N^*(C)$ 之间存在幂律关系：

$$N^*(C) = AC^\alpha$$

我们使用图 2 中的数据拟合 A 和 α。我们发现 $(\alpha, \mathrm{A}) = (0.53, 0.29)$；相应的拟合显示在图 3 中。将结果规模法则外推到 3.8×10^{25} FLOPs 建议在 16.55T 令牌上训练一个 402B 参数模型。

一个重要的观察是，随着计算预算的增加，IsoFLOPs 曲线在最小值附近变得更平。这意味着旗舰模型的性能对于模型大小和训练令牌之间的权衡的小变化相对稳健。基于这一观察，我们最终决定训练一个具有 4050 亿参数的旗舰模型。

预测下游任务的性能 我们使用结果中的计算最优模型来预测旗舰 Llama 3 模型在基准数据集上的性能。首先，我们线性相关基准中正确答案的（归一化的）负对数似然和训练 FLOPs。在这个分析中，我们只使用在上述数据混合上训练到 10^{22} FLOPs 的规模法则模型。接下来，我们使用规模法则模型和 Llama 2 模型建立对数似然和准确率之间的 S 形关系，这些模型是使用 Llama 2 数据混合和分词器训练的。我们在图 4 中展示了这个实验的结果（ARC 挑战基准）。我们发现这个两步规模法则预测，它在四个数量级上进行外推，相当准确：它只是略微低估了旗舰 Llama 3 模型的最终性能。

3.3 基础设施、扩展和效率

我们描述了支持 Llama 3 405B 大规模预训练的硬件和基础设施，并讨论了几项优化，这些优化带来了训练效率的提升。

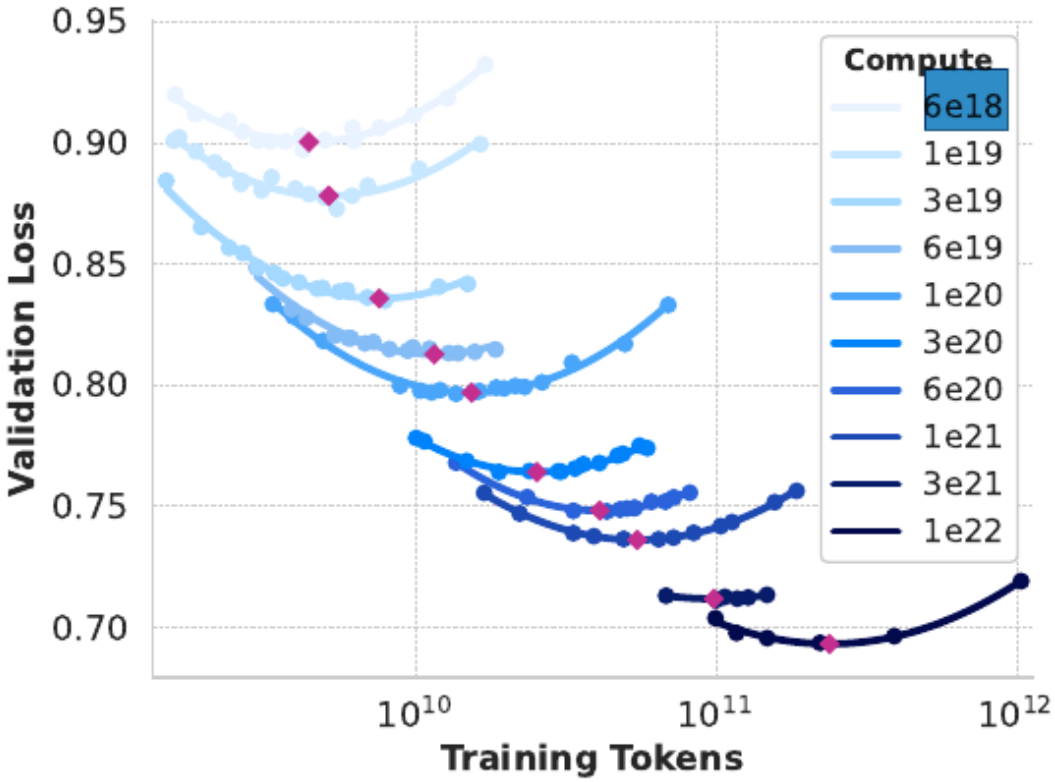

图 5: 图 2 在 6×10^{18} 到 10^{22} FLOPs 之间的规模法则 IsoFLOPs 曲线。损失是在一个保留的验证集上的负对数似然。我们使用二度多项式近似每个计算规模的测量值。

3.3.1 训练基础设施

Llama 1 和 2 模型是在 Meta 的 AI 研究超级集群（Lee 和 Sengupta，2022）上训练的。随着我们进一步扩展，Llama 3 的训练转移到了 Meta 的生产集群（Lee 等人，2024）。这个设置优化了生产级别的可靠性，这对于我们扩大训练规模至关重要。

计算 Llama 3 405B 在多达 16K 个 H100 GPU 上进行训练，每个 GPU 运行 700W TDP(Thermal Design Power)，配备 80GB HBM3，使用 Meta 的 Grand Teton AI 服务器平台（Matt Bowman，2022）。每个服务器配备八个 GPU 和两个 CPU。在服务器内，八个 GPU 通过 NVLink 连接。使用 MAST（Choudhury 等人，2024）调度训练作业，这是 Meta 的全球规模训练调度器。

存储 Tectonic（Pan 等人，2021），Meta 的通用分布式文件系统，用于构建 Llama 3 预训练的存储网络（Battey 和 Gupta，2024）。它提供了 7500 台配备 SSD 的服务器中的 240PB 存储空间，并支持持续吞吐量 2TB/s 和峰值吞吐量 7TB/s。一个主要挑战是支持高突发性的检查点写入，这些写入在短时间内饱和存储网络。检查点保存每个 GPU 的模型状态，范围从每个 GPU 的 1MB 到 4GB，用于恢复和调试。我们的目标是最小化检查点期间 GPU 的暂停时间，并增加检查点频率，以减少恢复后丢失的工作量。

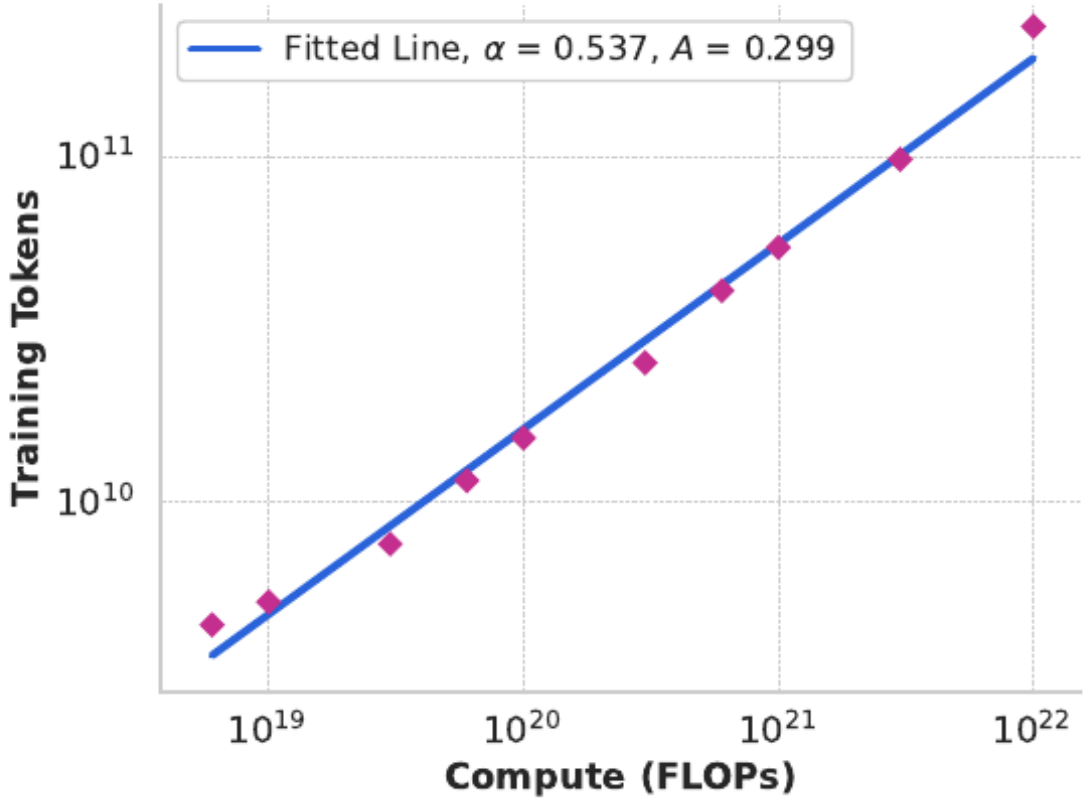

图 6: 图 3 被识别的计算最优模型中的训练令牌数量与预训练计算预算的函数关系。我们也包括了拟合的规模法则预测。计算最优模型对应于图 2 中的抛物线最小值。

网络 Llama 3 405B 使用了基于 Arista 7800 和 Minipack2 开放计算项目 4 OCP 机架交换机的 RDMA over Converged Ethernet (RoCE) 网络。Llama 3 家族的较小模型使用 Nvidia Quantum2 Infiniband 网络进行训练。RoCE 和 Infiniband 集群都在 GPU 之间利用 400Gbps 互连。尽管这些集群的底层网络技术存在差异，但我们调整了它们，以提供这些大型训练工作负载的等效性能。我们将更详细地介绍我们的 RoCE 网络，因为我们完全拥有其设计。

- **网络拓扑**我们的基于 RoCE 的 AI 集群由 24K GPU 连接，由三层 Clos 网络（Lee 等人，2024）组成。在底层，每个机架托管 16 个 GPU，分布在两台服务器上，并通过单个 Minipack2 机架顶部（ToR）交换机连接。在中间层，192 个这样的机架通过集群交换机连接，形成一个拥有 3072 GPU 的吊舱，具有完整的双工带宽，确保没有过订 (oversubscription)。在顶层，同一数据中心大楼内的八个这样的吊舱通过聚合交换机连接，形成一个 24K GPU 的集群。

- **负载均衡** LLM 训练产生难以使用传统方法（如等成本多路径（ECMP）路由）在所有可用网络路径上进行负载均衡的胖网络流。为解决这一挑战，我们采用了两种技术。首先，我们的集合库在两个 GPU 之间创建 16 个网络流，而不是仅仅一个，从而减少了每个流的流量，并为负载均衡提供

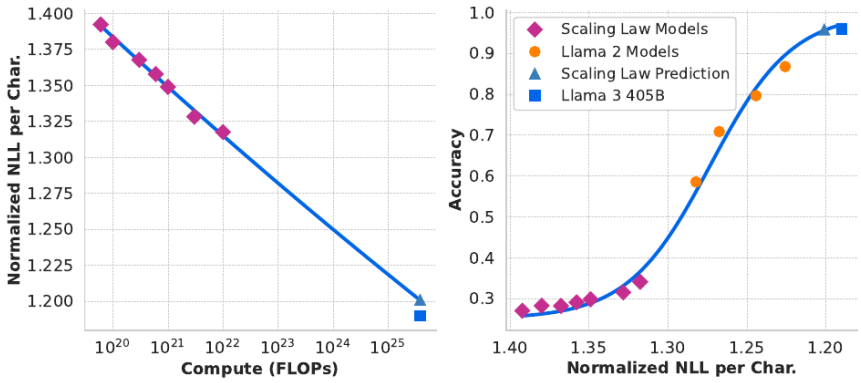

图 7: 图 4 ARC 挑战的规模法则预测。左侧：在 ARC 挑战基准测试中，正确答案的归一化负对数似然作为预训练 FLOPs 的函数。右侧：ARC 挑战基准测试的准确率作为正确答案的归一化负对数似然的函数。这种分析使我们能够在预训练开始之前预测模型在 ARC 挑战基准测试上的性能。详见正文。

了更多的流。其次，我们的增强 ECMP（E-ECMP）协议通过在 RoCE 数据包头中散列额外字段，有效地在不同网络路径上平衡这 16 个流。

- **拥塞控制**我们使用深缓冲交换机在主干（Gangidi 等人，2024）中容纳由集合通信模式引起的瞬态拥塞和缓冲。这种设置有助于限制由慢服务器引起的持续拥塞和网络反压的影响，这在训练中很常见。最后，通过 E-ECMP 的更好负载均衡显著降低了拥塞的可能性。通过这些优化，我们成功地运行了一个 24K GPU 集群，而无需使用传统的拥塞控制方法，如数据中心量化拥塞通知（DCQCN）。

3.3.2 模型扩展的并行性

为了扩展我们最大模型的训练，我们使用 4D 并行性——四种不同类型的并行方法的组合——来分片模型。这种方法有效地在许多 GPU 上分布计算，并确保每个 GPU 的模型参数、优化器状态、梯度和激活适合其 HBM。我们在图 5 中展示了 4D 并行性的实现。它结合了张量并行性（TP；Krizhevsky 等人，2012；Shoeybi 等人，2019；Korthikanti 等人，2023）、流水线并行性（PP；Huang 等人，2019；Narayanan 等人，2021；Lamy-Poirier，2023）、上下文并行性（CP；Liu 等人，2023a）和数据并行性（DP；Rajbhandari 等人，2020；Ren 等人，2021；Zhao 等人，2023b）。

张量并行性将个别权重张量分割成不同设备上的多个块。流水线并行性通过层垂直划分模型，使不同的设备可以并行处理完整模型流水线的不同阶段。上下文并行性将输入上下文分割成段，减少了非常长序列长度输入的内存瓶颈。我们使用完全分片的数据并行性（FSDP；Rajbhandari 等人，2020；Ren 等人，2021；Zhao 等人，2023b），它在实现数据并行性的同时分片模型、优化器和梯度，数据并行性在多个 GPU 上并行处理数据，并在每个训练步骤后同步。我们对 Llama 3 使用 FSDP 的方式是分片优化器状态和梯度，但对于模型分片我们没有在前向计算后重新分片，以避免在后向传递期间进行额外的全收集通信。

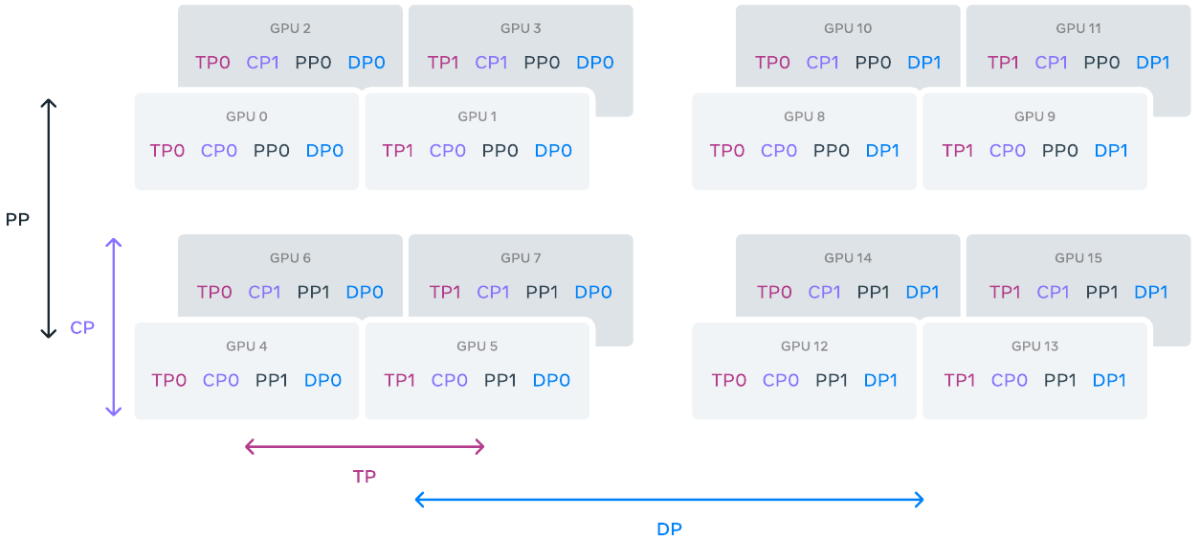

图 8: 图 5 4D 并行性的示意图。GPU 按照 [TP, CP, PP, DP] 的顺序被划分为并行组，其中 DP 代表 FSDP。在这个例子中，16 个 GPU 被配置为组大小为 |TP|=2, |CP|=2, |PP|=2, 和 |DP|=2。GPU 在 4D 并行中的位置被表示为一个向量，[D1, D2, D3, D4]，其中 Di 是第 i 个并行维度上的索引。在这个例子中，GPU0[TP0, CP0, PP0, DP0] 和 GPU1[TP1, CP0, PP0, DP0] 在同一个 TP 组中，GPU0 和 GPU2 在同一个 CP 组中，GPU0 和 GPU4 在同一个 PP 组中，GPU0 和 GPU8 在同一个 DP 组中。

GPU 利用率 通过仔细调整并行配置、硬件和软件，我们在表 4 所示的配置中实现了整体 BF16 模型 FLOPs 利用率（MFU；Chowdhery 等人，2023）为 38-43%。在 16K GPU 上，DP=128 的 MFU 略微下降到 41%，而 8K GPU 上，DP=64 的 MFU 为 43%，这是由于在训练期间需要保持全局批次中的令牌数量不变，因此每个 DP 组的批量大小较低。

GPUs	TP	CP	PP	DP	Seq. Len.	Batch size/DP	Tokens/Batch	TFLOPs/GPU	BF16 MFU
8,192	8	1	16	64	8,192	32	16M	430	43%
16,384	8	1	16	128	8,192	16	16M	400	41%
16,384	8	16	16	4	131,072	16	16M	380	38%

图 9: 表 4 Llama 3 405B 预训练各阶段的扩展配置和 MFU（模型 FLOPs 利用率）。详见文本和图 5，了解每种并行类型的描述。

流水线并行性改进 我们在使用现有实现时遇到了几个挑战：

- 批量大小限制。当前实现对每个 GPU 支持的批量大小有限制，要求它能够被流水线阶段的数量整除。例如，在图 6 中，流水线并行性的深度优先调度（DFS）要求 N = PP = 4，而广度优先调度

（BFS；Lamy-Poirier（2023））要求 N = M，其中 M 是微批次的总数，N 是同一阶段的前向或后向的连续微批次的数量。然而，预训练通常需要灵活性来调整批量大小。

- 内存不平衡。现有的流水线并行性实现导致资源消耗不平衡。第一阶段由于嵌入和预热微批次而消耗更多的内存。
- 计算不平衡。在模型的最后一层之后，我们需要计算输出和损失，使这个阶段成为执行延迟瓶颈。

为解决这些问题，我们修改了我们的流水线调度，如图 6 所示，它允许灵活设置 N——在这种情况下 N = 5，可以运行每个批次中的任意数量的微批次。这允许我们运行：(1) 在大规模时，比阶段数量少的微批次，当我们有批量大小限制；(2) 更多的微批次来隐藏点对点通信，找到 DFS 和 BFS 的最佳通信和内存效率之间的平衡点。为了平衡流水线，我们分别从第一阶段和最后阶段减少了一个 Transformer 层。这意味着第一阶段的第一个模型块只有嵌入，最后阶段的最后一个模型块只有输出投影和损失计算。为了减少流水线气泡，我们使用了一个交错调度（Narayanan 等人，2021）在 V 个流水线阶段上运行一个流水线等级。总体流水线气泡比率是 $\frac{PP-1}{V*M}$。此外，我们采用流水线中的异步点对点通信，这大大加快了训练速度，特别是在文档掩码引入额外计算不平衡的情况下。我们启用 TORCH_NCCL_AVOID_RECORD_STREAMS 来减少异步点对点通信的内存使用。最后，为了减少内存成本，基于详细的内存分配分析，我们主动释放了将来不会用于计算的张量，包括每个流水线阶段的输入和输出张量，这些张量将来不会用于计算。通过这些优化，我们能够在没有激活检查点的情况下预训练 Llama 3 在 8K 令牌序列上。

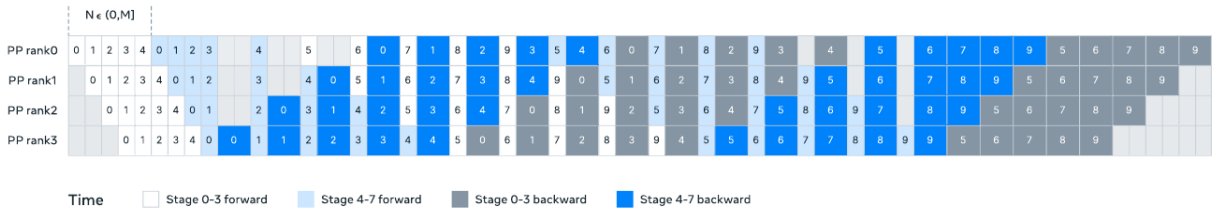

图 10: 图 6 Llama 3 中的流水线并行性示意图。流水线并行性将八个流水线阶段（0 到 7）划分到四个流水线等级（PP 等级 0 到 3）上，其中等级为 0 的 GPU 运行阶段 0 和 4，等级为 1 的 GPU 运行阶段 1 和 5，依此类推。彩色块（0 到 9）代表一系列微批次，其中 M 是微批次的总数，N 是同一阶段前向或后向的连续微批次的数量。我们的关键见解是使 N 可调。

长序列的上下文并行性 我们利用上下文并行性（CP）在扩展 Llama 3 的上下文长度时提高内存效率，并启用在长达 128K 的极长序列上的训练。在 CP 中，我们跨序列维度进行划分，具体来说，我们将输入序列划分为 2 × CP 块，以便每个 CP 等级接收两个块以更好地平衡负载。第 i 个 CP 等级接收了第 i 个和 (2 × CP − 1 − i)-th 块。

与现有的 CP 实现不同，后者在环状结构中重叠通信和计算（Liu 等人，2023a），我们的 CP 实现采用了基于 all-gather 的方法，我们首先 all-gather 关键（K）和值（V）张量，然后计算局部查询（Q）张量块的注意力输出。尽管 all-gather 通信延迟在关键路径上暴露出来，但我们仍然采用这种方法，主要有两个原因：(1) 在 all-gather 基础的 CP 注意力中，支持不同类型的注意力掩码（如文档掩码）更容易、更灵活；(2) 由于使用了 GQA（Ainslie 等人，2023），通信的 K 和 V 张量比 Q 张量小得多，因此暴

露的 all-gather 延迟很小。因此，注意力计算的时间复杂度比 all-gather 大一个数量级（$O(S^2)$ 与 O(S)，其中 S 表示全因果掩码中的序列长度），使 all-gather 开销微不足道。

网络感知并行配置 并行维度的顺序，[TP, CP, PP, DP]，针对网络通信进行了优化。最内层的并行需要最高的网络带宽和最低的延迟，因此通常限制在同一个服务器内。最外层的并行可以跨越多跳网络，并应容忍更高的网络延迟。因此，基于对网络带宽和延迟的要求，我们按照 [TP, CP, PP, DP] 的顺序放置并行维度。DP（即 FSDP）是最外层的并行，因为它可以通过异步预取分片模型权重和减少梯度来容忍更长的网络延迟。在避免 GPU 内存溢出的同时识别具有最小通信开销的最佳并行配置是具有挑战性的。我们开发了一个内存消耗估计器和性能预测工具，帮助我们探索各种并行配置，并有效地预测整体训练性能和识别内存差距。

数值稳定性 通过比较不同并行设置之间的训练损失，我们修复了几个影响训练稳定性的数值问题。为确保训练收敛，我们在多个微批次的后向计算中使用 FP32 梯度累积，并在 FSDP 中以 FP32 跨数据并行工作器减少梯度。对于在前向计算中多次使用的中间张量，例如视觉编码器输出，反向梯度也以 FP32 累积。

3.3.3 集体通信

我们 Llama 3 的集体通信库是基于 Nvidia 的 NCCL 库的一个分支，称为 NCCLX。NCCLX 显著提高了 NCCL 的性能，特别是在高延迟网络上。回想一下，平行维度的顺序是 [TP, CP, PP, DP]，其中 DP 对应于 FSDP。最外层的平行维度 PP 和 DP 可能通过多跳网络通信，延迟高达数十微秒。原始 NCCL 集合——FSDP 中的 all-gather 和 reduce-scatter，以及 PP 中的点对点——需要数据分块和分阶段数据复制。这种方法带来了几个效率问题，包括 (1) 需要在网络上交换大量小控制消息以促进数据传输，(2) 额外的内存复制操作，以及 (3) 使用额外的 GPU 周期进行通信。对于 Llama 3 训练，我们通过调整分块和数据传输以适应我们的网络延迟来解决这些效率问题的一个子集，这些延迟对于大型集群可能高达数十微秒。我们还允许小控制消息以更高的优先级在我们的网络上传输，特别是避免在深缓冲核心交换机中被头部阻塞。我们为未来 Llama 版本进行的持续工作涉及在 NCCLX 中进行更深入的更改，以全面解决前述所有问题。

3.3.4 可靠性和操作挑战

16K GPU 训练的复杂性和潜在故障场景超过了我们运营的规模大得多的 CPU 集群。此外，训练的同步性质使其对故障的容忍度较低——单个 GPU 故障可能需要整个作业重新启动。尽管存在这些挑战，对于 Llama 3，我们实现了超过 90% 的有效训练时间，同时支持自动化集群维护，如固件和 Linux 内核升级（Vigraham 和 Leonhardi，2024），这导致每天至少有一次训练中断。有效训练时间衡量的是过去时间中用于有用训练的时间。

在预训练的 54 天快照期间，我们共经历了 466 次作业中断。其中，47 次是计划内的中断，由于自动化维护操作，如固件升级或操作员启动的操作，如配置或数据集更新。其余 419 次是意外的中断，在表 5 中进行了分类。约 78% 的意外中断归因于确认的硬件问题，如 GPU 或主机组件故障，或疑似与硬件相关的问题，如沉默数据损坏和计划外的单个主机维护事件。GPU 问题是最大类别，占所有意外问题的 58.7%。尽管失败数量众多，但在此期间仅需要三次重要的手动干预，其余问题由自动化处理。

Component	Category	Interruption Count	% of Interruptions
Faulty GPU	GPU	148	30.1%
GPU HBM3 Memory	GPU	72	17.2%
Software Bug	Dependency	54	12.9%
Network Switch/Cable	Network	35	8.4%
Host Maintenance	Unplanned Maintenance	32	7.6%
GPU SRAM Memory	GPU	19	4.5%
GPU System Processor	GPU	17	4.1%
NIC	Host	7	1.7%
NCCL Watchdog Timeouts	Unknown	7	1.7%
Silent Data Corruption	GPU	6	1.4%
GPU Thermal Interface + Sensor	GPU	6	1.4%
SSD	Host	3	0.7%
Power Supply	Host	3	0.7%
Server Chassis	Host	2	0.5%
IO Expansion Board	Host	2	0.5%
Dependency	Dependency	2	0.5%
CPU	Host	2	0.5%
System Memory	Host	2	0.5%

图 11: 表 5 在 Llama 3 405B 预训练的 54 天期间，意外中断的根本原因分类。约 78% 的意外中断归因于确认或疑似的硬件问题。

为了增加有效训练时间，我们减少了作业启动和检查点时间，并开发了快速诊断和问题解决工具。我们广泛使用 PyTorch 内置的 NCCL 飞行记录器（Ansel 等人，2024），这是一个功能，可以捕获集体元数据和堆栈跟踪到环形缓冲区，因此允许我们快速诊断大规模的挂起和性能问题，特别是与 NCCLX 有关的问题。利用这个，我们有效地记录了每个通信事件和每个集体操作的持续时间，并在 NCCLX 看门狗或心跳超时时自动转储跟踪数据。我们通过在线配置更改（Tang 等人，2015）根据需要在生产中选择性地启用更多计算密集型的跟踪操作和元数据收集，无需代码发布或作业重启。

在我们网络中混合使用 NVLink 和 RoCE 使得大规模训练中的问题调试变得复杂。通过 NVLink 的数据传输通常通过 CUDA 内核发出的加载/存储操作进行，远程 GPU 或 NVLink 连接的故障通常表现为 CUDA 内核内的加载/存储操作停滞，而没有返回清晰的错误代码。NCCLX 通过与 PyTorch 的紧密共同设计，提高了故障检测和定位的速度和准确性，允许 PyTorch 访问 NCCLX 的内部状态并跟踪相关信息。虽然由于 NVLink 故障引起的停滞无法完全防止，但我们的系统监控通信库的状态，并在检测到此类停滞时自动超时。此外，NCCLX 跟踪每个 NCCLX 通信的内核和网络活动，并提供了失败 NCCLX 集体内部状态的快照，包括所有等级之间已完成和待定的数据传输。我们分析这些数据来调试 NCCLX 扩展问题。

有时，硬件问题可能导致仍在运行但速度较慢的落后者，这些落后者很难检测到。即使一个落后者也可以减慢数千其他 GPU 的速度，通常表现为通信功能正常但速度较慢。我们开发了工具，以优先处理选定进程组中可能存在问题的通信。通过仅调查少数主要嫌疑人，我们通常能够有效地识别落后者。

一个有趣的观察是环境因素对大规模训练性能的影响。对于 Llama 3 405B，我们注意到基于一天中时间的日波动 1-2% 的吞吐量变化。这种波动是由于中午温度较高影响 GPU 动态电压和频率缩放。在训练期间，成千上万的 GPU 可能同时增加或减少功耗，例如，由于所有 GPU 等待检查点或集体通信完成，或整个训练作业的启动或关闭。当这种情况发生时，它可能导致数据中心的功耗瞬间波动数十兆瓦，挑战电网的极限。随着我们为未来更大规模的 Llama 模型扩展训练，这是我们面临的一个持续挑战。

3.4 训练配方

用于预训练 Llama 3 405B 的配方包括三个主要阶段：(1) 初始预训练，(2) 长上下文预训练，以及 (3) 退火。下面分别描述这三个阶段。我们使用类似的配方来预训练 8B 和 70B 模型。

3.4.1 初始预训练

我们使用余弦学习率计划对 Llama 3 405B 进行预训练，峰值学习率为 8×10^{-5}，线性预热 8000 步，然后在 120 万训练步骤中衰减到 8×10^{-7}。我们在训练初期使用较小的批量大小以提高训练稳定性，并随后增加批量大小以提高效率。具体来说，我们最初使用 4M 令牌的批量大小和长度为 4,096 的序列，然后在预训练了 2.52 亿令牌后，将这些值加倍到 8M 令牌，长度为 8,192 的序列。我们在预训练了 2.87T 令牌后再次将批量大小加倍到 16M。我们发现这种训练配方非常稳定：我们观察到的损失尖峰很少，并且不需要干预来纠正模型训练发散。

调整数据混合 在训练期间，我们对预训练数据混合进行了几次调整，以提高模型在特定下游任务上的性能。特别是，我们增加了非英语数据的百分比，以提高 Llama 3 的多语言性能。我们还上采样数学数据以提高模型的数学推理性能，我们在预训练的后期增加了更多最新的网络数据以推进模型的知识截止日期，并且我们对后来被识别为质量较低的预训练数据子集进行了下采样。

3.4.2 长上下文预训练

在预训练的最后阶段，我们对长序列进行训练，以支持高达 128K 令牌的上下文窗口。我们不早点在长序列上训练，因为在自注意力层中的计算在序列长度上呈二次方增长。我们逐步增加支持的上下文长度，预训练直到模型成功适应增加的上下文长度。我们通过测量 (1) 模型在短上下文评估上的性能是否完全恢复，以及 (2) 模型是否完美解决了长达该长度的 “大海捞针” 任务，来评估成功的适应。在 Llama 3 405B 预训练中，我们从原始的 8K 上下文窗口开始，逐步增加了六个阶段的上下文长度，最后达到最终的 128K 上下文窗口。这个阶段的长上下文预训练使用了大约 8000 亿训练令牌。

3.4.3 退火

在最后 4000 万个训练令牌的预训练期间，我们线性地将学习率退火到 0，同时保持 128K 令牌的上下文长度。在退火阶段，我们还调整了数据混合，以对非常高质量数据源进行上采样；见第 3.1.3 节。最后，我们在退火期间计算模型检查点的平均值（Polyak 平均值），以产生最终的预训练模型。

4 后训练处理

我们通过应用多轮后训练处理，或在预训练检查点的基础上，通过人类反馈（Ouyang 等人，2022 年；Rafailov 等人，2024 年）来调整 Llama 3 模型。每一轮后训练处理包括监督式微调（SFT）和直接偏好

优化（DPO；Rafailov 等人，2024 年），这些优化是在通过人工标注或合成生成的示例上进行的。我们在第4.1节和第??节中分别描述了我们的后训练建模和数据方法。我们还在第??节中详细说明了定制的数据策划策略，以改进推理、编码、事实性、多语种、工具使用、长文本和精确指令遵循。

4.1 建模

我们的后训练策略的核心是奖励模型和语言模型。我们首先使用人类标注的偏好数据（见第 4.1.2 节）在预训练检查点之上训练奖励模型。然后我们通过监督微调（SFT；见第 4.1.3 节）微调预训练的检查点，并通过直接偏好优化（DPO；见第 4.1.4 节）进一步调整检查点。此过程如图 12 所示。除非另有说明，我们的建模过程适用于 Llama 3 405B，为简化起见，我们将 Llama 3 405B 称为 Llama 3。

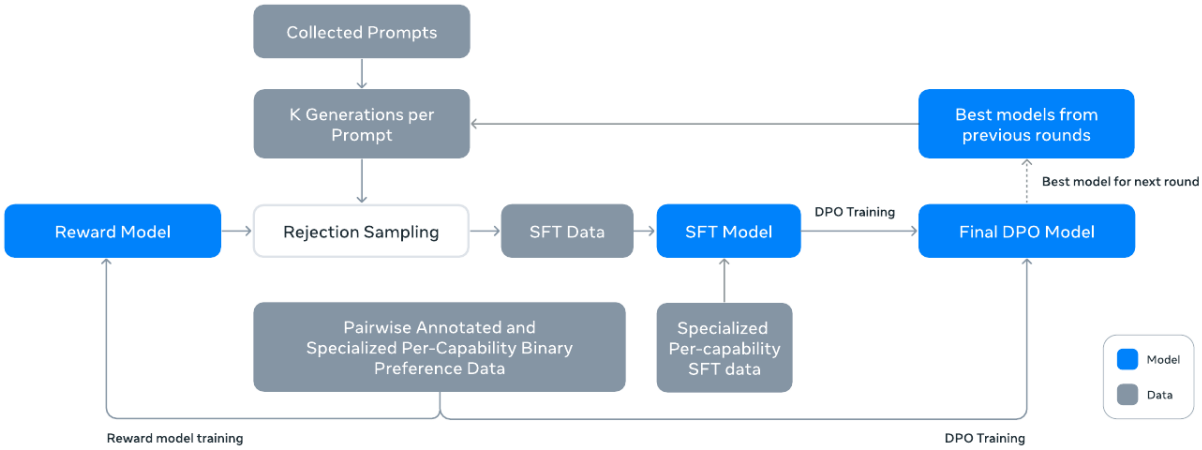

图 12: 展示了针对 Llama 3 的整体后训练方法。我们的后训练策略包括拒绝采样、监督式微调和直接偏好优化。详情见文本。

4.1.1 聊天对话格式

为了调整 LLMs 进行人机互动，我们需要为模型定义一个聊天对话协议，以便模型理解人类指令并执行对话任务。与其前身相比，Llama 3 具有新的功能，如工具使用（见第 ?? 节），这可能需要在单个对话回合内生成多条消息并将其发送到不同位置（例如用户、ipython）。为了支持这一点，我们设计了一个新的多消息聊天协议，该协议使用各种特殊的头标和终止 tokens。头标 tokens 用于指示对话中每条消息的来源和目的地。同样，终止 tokens 指示何时该轮换人类和 AI 说话。

4.1.2 奖励建模

我们在预训练检查点之上训练一个覆盖不同功能的奖励模型（RM）。训练目标与 Llama 2 相同，只是我们删除了损失中的边距项，因为我们观察到数据扩展后改进效果逐渐减弱。按照 Llama 2 的做法，我们在过滤掉具有类似响应的样本后，使用所有偏好数据进行奖励建模。除了标准的（选择的，拒绝的）响应偏好对之外，注释者还为某些提示创建了第三种“编辑响应”，即对偏好对中的选择响应进一步编辑

以进行改进（见第 **??** 节）。因此，每个偏好排名样本有两个或三个具有明确排名的响应（编辑 > 选择 > 拒绝）。在训练期间，我们将提示和多个响应连接成一行，响应随机打乱。这是将响应放在单独行中并计算分数的标准场景的近似，但在我们的消融实验中，这种方法提高了训练效率而不会降低准确性。

4.1.3 监督微调

然后使用奖励模型对我们的人类标注提示进行拒绝采样，详情见第 **??** 节。结合这些拒绝采样数据和其他数据源（包括合成数据），我们使用标准的交叉熵损失在目标 tokens 上微调预训练语言模型（同时在提示 tokens 上掩蔽损失）。有关数据混合的更多细节见第 **??** 节。我们将这一阶段称为监督微调（SFT；Wei 等，2022a；Sanh 等，2022；Wang 等，2022b），尽管许多训练目标是模型生成的。我们的最大模型在 8.5K 到 9K 步期间以 1×10^{-5} 的学习率进行微调。我们发现这些超参数设置在不同轮次和数据混合中效果很好。

4.1.4 直接偏好优化

我们进一步通过直接偏好优化（DPO；Rafailov 等，2024）训练我们的 SFT 模型，以进行人类偏好对齐。对于训练，我们主要使用从前几轮对齐中表现最佳的模型收集的最新偏好数据批次。因此，我们的训练数据更符合每轮优化的策略模型的分布。我们还探索了 PPO（Schulman 等，2017）等政策内算法，但发现 DPO 对大规模模型来说计算需求更少，并且在指令遵循基准测试（如 IFEval；Zhou 等，2023）中表现更好。对于 Llama 3，我们使用 1×10^{-5} 的学习率，并将 β 超参数设置为 0.1。此外，我们对 DPO 进行了以下算法修改：

- **在 DPO 损失中屏蔽格式化 tokens**：我们在选择和拒绝的响应中屏蔽包括头标和终止 tokens 在内的特殊格式化 tokens，以稳定 DPO 训练。我们观察到，这些 tokens 对损失的贡献可能会导致不希望的模型行为，如尾部重复或突然生成终止 tokens。我们假设这是由于 DPO 损失的对比性质——在选择和拒绝响应中同时存在的共同 tokens 会导致模型需要同时增加和减少这些 tokens 的可能性，从而导致学习目标冲突。

- **用 NLL 损失正则化**：我们在选择序列上添加一个额外的负对数似然（NLL）损失项，缩放系数为 0.2，类似于 Pang 等（2024）。这有助于通过保持生成的格式化并防止选择响应的对数概率下降来进一步稳定 DPO 训练（Pang 等，2024；Pal 等，2024）。

4.1.5 模型平均

最后，我们在每个 RM、SFT 或 DPO 阶段，使用不同版本的数据或超参数进行实验得到的模型进行平均（Izmailov 等人，2019 年；Wortsman 等人，2022 年；Li 等人，2022 年）。

4.1.6 迭代轮次

遵循 Llama 2 的做法，我们应用上述方法进行六轮迭代。在每个周期中，我们收集新的偏好标注和 SFT 数据，从最新模型中采样合成数据。

4.2 后训练数据

后训练数据的组成在语言模型的有用性和行为上起着至关重要的作用。在本节中，我们讨论了我们的人工标注程序和偏好数据收集（4.2.1节），SFT 数据的组成（4.2.2节），以及数据质量控制和清洗的方法（4.2.3节）。

4.2.1 偏好数据

我们的偏好数据注释过程与 Llama 2 类似。我们在每一轮后部署多个模型进行注释，并为每个用户提示从两个不同模型中抽取两个响应。这些模型可以采用不同的数据混合和对齐配方进行训练，允许不同的能力强度（例如，编码专业知识）和增加数据多样性。我们要求注释者根据他们对所选响应相对于被拒绝响应的偏好程度，将其分类为四个等级之一：明显更好、更好、略好或稍微更好。我们还在偏好排名后加入一个编辑步骤，鼓励注释者进一步改进首选响应。注释者可以直接编辑所选响应，或者用反馈提示模型以改进自己的响应。因此，我们的偏好数据中有部分有三个响应的排名（编辑 > 选择 > 拒绝）。

Dataset	% of comparisons	Avg. # turns per dialog	Avg. # tokens per example	Avg. # tokens in prompt	Avg. # tokens in response
General English	81.99%	4.1	1,000.4	36.4	271.2
Coding	6.93%	3.2	1,621.0	113.8	462.9
Multilingual	5.19%	1.8	1,299.4	77.1	420.9
Reasoning and tools	5.89%	1.6	707.7	46.6	129.9
Total	100%	3.8	1,041.6	44.5	284.0

图 13: 人类偏好数据的统计信息。我们列出了用于 Llama 3 对齐的内部收集的人类偏好数据的统计信息。我们要求注释者与模型进行多轮对话，并在每一轮中对响应进行比较。在后处理中，我们将每个对话拆分为多个在轮次级别的示例。每个示例包括一个提示（如果可用，包括之前的对话）和一个响应（例如，选择或拒绝的响应）。

在表13中，我们报告了我们用于 Llama 3 训练的偏好注释的统计信息。通用英语涵盖了多个子类别，如基于知识的问答或精确指令遵循，这些超出了特定能力的范围。与 Llama 2 相比，我们观察到提示和响应的平均长度有所增加，这表明我们正在训练 Llama 3 处理更复杂的任务。此外，我们实施了质量分析和人工评估流程，以严格评估收集的数据，使我们能够完善我们的提示并为注释者提供系统化、可操作的反馈。例如，随着 Llama 3 在每一轮后改进，我们相应地增加提示的复杂性，以针对模型落后的领域。在每一轮后训练，我们使用当时可用的所有偏好数据进行奖励建模，而仅使用各种能力的最新批次进行 DPO 训练。对于奖励建模和 DPO，我们使用标记为所选响应明显更好或比拒绝的对应物更好的样本进行训练，并丢弃响应相似的样本。

4.2.2 SFT 数据

我们的微调数据主要由以下来源组成：

- 来自我们人工标注收集的提示，带有拒绝采样的响应

- 针对特定能力的合成数据（详见??节）
- 少量人工策划的数据（详见??节）

随着我们的后训练轮次的进展，我们开发了更强大的 Llama 3 变体，我们用它们来收集涵盖广泛复杂能力的更大数据集。在本节中，我们将讨论拒绝采样程序的细节和我们最终 SFT 数据混合的总体组成。

拒绝采样 在拒绝采样（RS）期间，对于在人工标注中收集的每个提示（4.2.1节），我们从最新的聊天模型策略中采样 K（通常在 10 到 30 之间）个输出（通常是上一次后训练迭代中表现最佳的检查点，或者是特定能力的最优秀检查点），并使用我们的奖励模型选择最佳候选，这与 Bai 等人（2022 年）的方法一致。在后训练的后期轮次中，我们引入系统提示，引导 RS 响应符合期望的语调、风格或格式，这些可能因不同能力而异。

为了提高拒绝采样的效率，我们采用了 PagedAttention（Kwon 等人，2023 年）。PagedAttention 通过动态键值缓存分配增强了内存效率。它通过根据当前缓存容量动态调度请求来支持任意输出长度。不幸的是，这在内存不足时会带来交换开销的风险。为了消除这种交换开销，我们定义了最大输出长度，并仅在有足够的内存来容纳该长度的输出时才执行请求。PagedAttention 还使我们能够跨所有相应的输出共享提示的键值缓存页面。这共同导致了拒绝采样期间吞吐量的 2 倍以上的提高。

Dataset	% of examples	Avg. # turns	Avg. # tokens	Avg. # tokens in context	Avg. # tokens in final response
General English	52.66%	6.3	974.0	656.7	317.1
Code	14.89%	2.7	753.3	378.8	374.5
Multilingual	3.01%	2.7	520.5	230.8	289.7
Exam-like	8.14%	2.3	297.8	124.4	173.4
Reasoning and tools	21.19%	3.1	661.6	359.8	301.9
Long context	0.11%	6.7	38,135.6	37,395.2	740.5
Total	100%	4.7	846.1	535.7	310.4

图 14: SFT 数据的统计信息。我们列出了用于 Llama 3 对齐的内部收集的 SFT 数据。每个 SFT 示例由一个上下文（即，除最后一个之外的所有对话轮次）和一个最终响应组成。

总体数据组成 表14显示了我们 “有用性” 混合中每个广泛类别的数据统计信息。虽然 SFT 和偏好数据包含重叠的领域，但它们以不同的方式策划，产生了不同的计数统计数据。在4.2.3节中，我们将描述对我们的数据样本进行主题、复杂性和质量分类的技术。在每一轮后训练，我们仔细调整我们的整体数据混合，以调整在广泛基准测试中的性能。我们最终的数据混合在一些高质量资源上多次循环，并对其他资源进行下采样。

4.2.3 数据处理和质量控制

鉴于我们的大部分训练数据是模型生成的，因此需要仔细清洗和质量控制。

数据清洗 在早期轮次中，我们观察到数据中存在一些不希望出现的模式，例如过度使用表情符号或感叹号。因此，我们实施了一系列基于规则的数据移除和修改策略，以过滤或清理有问题的数据。例如，为了减轻过于道歉的语调问题，我们识别过度使用的短语（如 “I’ m sorry” 或 “I apologize”），并仔细平衡这些样本在我们数据集中的比例。

数据修剪 我们还应用一系列基于模型的技术来移除低质量的训练样本并提高整体模型性能：

- **主题分类**：我们首先将 Llama 3 8B 微调为一个主题分类器，并在所有数据上进行推断，将其分类为粗粒度的桶（“数学推理”）和细粒度的桶（“几何和三角学”）。
- **质量评分**：我们使用奖励模型和基于 Llama 的信号为每个样本获得一个质量分数。对于基于 RM 的分数，我们认为在 RM 分数中处于上四分位数的数据为高质量。对于基于 Llama 的分数，我们提示 Llama 3 检查点对每个样本在一般英语数据上进行三点量表评分（准确性、指令遵循和语调/展示）以及对编码数据进行两点量表评分（错误识别和用户意图），并认为获得最高分数的样本为高质量。RM 和基于 Llama 的分数有很高的不一致率，我们发现结合这些信号在我们的内部测试集上获得最佳召回率。最终，我们选择被 RM 或基于 Llama 的过滤器标记为高质量的示例。
- **难度评分**：因为我们也对优先考虑对模型更复杂的示例感兴趣，我们使用两个难度度量对数据进行评分：Instag（Lu 等人，2023 年）和基于 Llama 的评分。对于 Instag，我们提示 Llama 3 70B 对 SFT 提示进行意图标记，其中更多的意图意味着更复杂。我们还提示 Llama 3 在三点量表上测量对话的难度（Liu 等人，2024 年 c）。
- **语义去重**：最后，我们执行语义去重（Abbas 等人，2023 年；Liu 等人，2024 年 c）。我们首先使用 RoBERTa（Liu 等人，2019b）对完整对话进行聚类，并在每个聚类内按质量分数 × 难度分数排序。然后我们通过迭代所有排序的示例，进行贪婪选择，并只保留那些与聚类中到目前为止看到的示例的最大余弦相似度小于阈值的示例。

4.3 能力

我们特别强调为提高特定能力的性能所做的努力，例如代码（4.3.1节）、多语言（**??**节）、数学和推理（4.3.3节）、长文本（4.3.4节）、工具使用（**??**节）、事实性（**??**节）和可引导性（**??**节）。

4.3.1 代码

自从 Copilot 和 Codex（Chen 等人，2021 年）发布以来，用于代码的大型语言模型（LLMs）受到了广泛关注。开发者现在广泛使用这些模型来生成代码片段、调试、自动化任务和提高代码质量。对于 Llama 3，我们的目标是提高以下高优先级编程语言的代码生成、文档编写、调试和审查能力：Python、Java、Javascript、C/C++、Typescript、Rust、PHP、HTML/CSS、SQL、bash/shell。在这里，我们展示了通过训练代码专家、为 SFT 生成合成数据、通过系统提示引导改进格式化以及创建质量过滤器从我们的训练数据中移除不良样本来提高这些编码能力的工作。

专家训练 我们训练了一个代码专家，用于在随后的训练轮次中收集高质量的代码人工标注。这是通过分支主预训练运行并在主要是（>85%）代码数据的 1T 令牌混合上继续预训练来实现的。继续在特定领域的数据上进行预训练已被证明对提高特定领域的性能有效（Gururangan 等人，2020 年）。我们遵循与 CodeLlama（Rozière 等人，2023 年）类似的配方。在训练的最后几千步中，我们执行长文本微调

(LCFT), 以将专家的上下文长度扩展到 16K 令牌的高质量混合存储库级代码数据。最后, 我们遵循**??**节中描述的类似后训练建模配方来对齐这个模型, 只是 SFT 和 DPO 数据混合主要针对代码。这个模型也用于代码提示的拒绝采样（4.2.2节）。

合成数据生成 在开发过程中, 我们确定了代码生成中的一些关键问题, 包括难以遵循指令、代码语法错误、错误的代码生成以及难以修复错误。虽然理论上密集的人工标注可以解决这些问题, 但合成数据生成提供了一种成本更低、规模更大、不受注释者专业水平限制的补充方法。因此, 我们使用 Llama 3 和代码专家生成了大量合成 SFT 对话。

我们描述了生成合成代码数据的三种高层方法。总共, 我们生成了超过 270 万个合成示例, 这些示例在 SFT 中使用。

1. 合成数据生成: 执行反馈 8B 和 70B 模型在训练时使用更大、更有能力模型生成的数据时显示出显著的性能提升。然而, 我们最初的实验表明, 训练 Llama 3 405B 使用自己生成的数据是没有帮助的（甚至可能降低性能）。为了解决这个限制, 我们引入执行反馈作为真实来源, 使模型能够从错误中学习并保持在正确的轨道上。特别是, 我们使用以下流程生成了大约一百万个合成编码对话的大型数据集:

- **问题描述生成**: 首先, 我们生成了大量涵盖多样化主题的编程问题描述, 包括长尾分布中的主题。为了实现这种多样性, 我们从各种来源随机采样代码片段, 并提示模型根据这些示例生成编程问题。这使我们能够利用广泛的主题, 并创建一套全面的问题描述（Wei 等人, 2024 年）。
- **解决方案生成**: 然后, 我们提示 Llama 3 用给定的编程语言解决每个问题。我们观察到, 在提示中添加通用的良好编程规则可以提高生成的解决方案质量。此外, 我们发现要求模型在注释中解释其思维过程也是有帮助的。
- **正确性分析**: 生成解决方案后, 至关重要的是要认识到其正确性并不是保证的, 将不正确的解决方案包括在微调数据集中可能会损害模型的质量。虽然我们不保证完全正确, 但我们开发了近似它的方法。为此, 我们从生成的解决方案中提取源代码, 并应用了静态和动态分析技术的组合来测试其正确性, 包括:
  - **静态分析**: 我们将所有生成的代码通过解析器和 linter 运行, 以确保语法正确性, 捕捉诸如语法错误、使用未初始化的变量或未导入的函数、代码风格问题、类型错误等错误。
  - **单元测试生成和执行**: 对于每个问题和解决方案, 我们提示模型生成单元测试, 在容器化环境中与解决方案一起执行, 捕捉运行时执行错误和一些语义错误。
- **错误反馈和迭代自我纠正**: 当解决方案在任何步骤失败时, 我们提示模型进行修订。提示包括原始问题描述、有缺陷的解决方案以及来自解析器/linter/测试器的反馈（stdout、stderr 和返回代码）。在单元测试执行失败后, 模型可以修复代码以通过现有测试, 或者修改其单元测试以适应生成的代码。只有通过所有检查的对话才包含在最终数据集中, 用于监督式微调（SFT）。值得注意的是, 我们观察到大约 20% 的解决方案最初是不正确的, 但自我纠正了, 这表明模型从执行反馈中学习并提高了其性能。

2. 合成数据生成: 编程语言翻译 我们观察到主要编程语言（例如, Python/C++）和不太常见的语言（例如, Typescript/PHP）之间的性能差距。这并不奇怪, 因为我们对不太常见的编程语言的训练数据较少。为了缓解这个问题, 我们通过将常见编程语言的数据翻译成不太常见的语言来补充我们现有的数据

(类似于 Chen 等人（2023 年）在推理背景下的做法)。这是通过提示 Llama 3 并确保通过语法解析、编译和执行来保证质量来实现的。图15展示了一个从 Python 翻译成 PHP 的合成代码示例。这显著提高了不太常见语言的性能，如 MultiPL-E（Cassano 等人，2023 年）基准测试所测量的。

```python
def gushti_cdi():
    n = int(input())
    arr = list(map(int, input().split()))
    points = 0
    for i in range(n):
        if arr[i] == 0 and i != 0:
            idx = arr.index(max(arr[:i]))
            points += arr[idx]
            arr[idx] = 0
    return points

for _ in range(int(input())):
    print(gushti_cdi())
```

```php
<?php
function gushti_cdi() {
    $n = (int)fgets(STDIN);
    $arr = explode(" ", fgets(STDIN));
    $points = 0;
    for ($i = 0; $i < $n; $i++) {
        if ($arr[$i] == 0 && $i != 0) {
            $maxVal = max(array_slice($arr, 0, $i));
            $idx = array_search($maxVal, $arr);
            $points += $arr[$idx];
            $arr[$idx] = 0;
        }
    }
    return $points;
}

$t = (int)fgets(STDIN);
for ($i = 0; $i < $t; $i++) {
    echo gushti_cdi() . "\n";
}
?>
```

图 15: 代码翻译示例。我们展示了使用 Llama 3 将 Python 代码（左侧）翻译成 PHP 代码（右侧）的例子，以增加我们的 SFT 数据集包含更广泛的编程语言。

3. 合成数据生成：回译 为了提高某些编码能力（例如，文档编写、解释）的性能，其中执行反馈对于确定质量的有用性较低，我们采用了一种替代的多步骤方法。使用这种方法，我们生成了大约 120 万个与代码解释、生成、文档编写和调试相关的合成对话。从我们的预训练数据中以各种语言编写的代码片段开始：

- **生成**：我们提示 Llama 3 生成代表我们目标能力的数据（例如，我们为代码片段添加注释和文档字符串，或者我们要求模型解释一段代码)。
- **回译**：然后，我们提示模型将合成生成的数据“回译”到原始代码（例如，我们提示模型仅根据其文档生成代码，或者我们要求模型仅根据其解释生成代码)。
- **过滤**：使用原始代码作为参考，我们提示 Llama 3 确定输出的质量（例如，我们问模型回译的代码与原始代码的忠实度如何)。然后我们在 SFT 中使用具有最高自我验证分数的生成示例。

拒绝采样过程中的系统提示引导 在拒绝采样过程中，我们使用特定于代码的系统提示来提高代码的可读性、文档编写、彻底性和特异性。回想一下，4.2.2节中的数据用于微调语言模型。图16展示了一个系统提示如何帮助提高生成的代码质量的示例——它添加了必要的注释，使用了更信息丰富的变量名，节省了内存等。

使用执行和模型作为裁判信号过滤训练数据 如**??**节所述，我们偶尔在我们的拒绝采样数据中遇到质量问题，例如包含错误的代码块。在我们的拒绝采样数据中检测这些问题并不像在我们的合成代码数据中那样直接，因为拒绝采样的响应通常包含自然语言和代码的混合，而代码可能并不总是预期为可执行的(例如，用户提示可能明确要求伪代码或仅对可执行程序的非常小的片段进行编辑)。为了解决这个问题，我们采用了“模型作为裁判”的方法，早期版本的 Llama 3 根据两个标准评估并分配一个二进制（0/1）

```
public static int ClimbStairs(int n)
{
    if (n == 1)
    {
        return 1;
    }

    if (n == 2)
    {
        return 2;
    }

    int[] dp = new int[n + 1];
    dp[1] = 1;
    dp[2] = 2;

    for (int i = 3; i <= n; i++)
    {
        dp[i] = dp[i - 1] + dp[i - 2];
    }

    return dp[n];
}
```

```
public static int ClimbStairs(int n)
{
    // Base cases
    if (n == 1) return 1;
    if (n == 2) return 2;

    // Initialize variables to store the number
of ways to climb
    int prev = 1;
    int curr = 2;

    // Calculate the number of ways to climb for
n > 2
    for (int i = 3; i <= n; i++)
    {
        int temp = curr;
        curr = prev + curr;
        prev = temp;
    }

    return curr;
}
```

图 16: 使用系统提示提高生成代码的质量。左侧：没有使用系统提示。右侧：使用了系统提示。

分数：代码正确性和代码风格。我们只保留那些获得 2 分完美分数的样本。最初，这种严格的过滤导致了下游基准性能的回归，主要是因为它不成比例地移除了具有挑战性的提示的示例。为了抵消这一点，我们有策略地修订了一些被归类为最具挑战性的编码数据的响应，直到它们满足基于 Llama 的“模型作为裁判”的标准。通过完善这些具有挑战性的问题，编码数据在质量和难度之间实现了平衡，从而实现了最佳的下游性能。

4.3.2 多语言性

我们描述了如何提高 Llama 3 的多语言能力，包括训练一个专门处理更多多语言数据的专家、为德语、法语、意大利语、葡萄牙语、印地语、西班牙语和泰语收集和生成高质量的多语言指令调整数据，以及应对多语言语言引导的具体挑战，以提高我们模型的整体性能。

专家训练 我们的 Llama 3 预训练数据混合包含了明显多于非英语令牌的英语令牌。为了收集非英语语言中更高质量的人工标注，我们通过分支预训练运行并在由 90% 多语言令牌组成的数据混合上继续预训练来训练一个多语言专家。然后，我们按照**??**节对这位专家进行后训练的调整。这个专家模型随后被用来在非英语语言中收集更高质量的注释，直到预训练完全完成。

多语言数据收集 我们的多语言 SFT 数据主要来源于以下描述的来源。总体分布是 2.4% 的人工标注，44.2% 来自其他 NLP 任务的数据，18.8% 的拒绝采样数据，和 34.6% 的翻译推理数据。

- **人工标注**：我们从语言学家和母语者那里收集高质量的、人工标注的数据。这些注释主要由代表现实世界用例的开放式提示组成。
- **来自其他 NLP 任务的数据**：为了进一步增强，我们使用来自其他任务的多语言训练数据，并将其改写成对话格式。例如，我们使用来自 exams-qa（Hardalov 等人，2020 年）和 Conic10k（Wu 等人，2023 年）的数据。为了提高语言对齐，我们还使用来自 GlobalVoices（Prokopidis 等人，2016 年）和 Wikimedia（Tiedemann，2012 年）的平行文本。我们使用基于 LID 的过滤和 Blaser2.0（Seamless Communication 等人，2023 年）来移除低质量数据。对于平行文本数据，我们不是直接使用双语

文本对，而是应用了受 Wei 等人（2022a）启发的多语言模板，以更好地模拟翻译和语言学习场景中的真实生活对话。

- **拒绝采样数据**：我们在人工标注的提示上应用拒绝采样来生成用于微调的高质量样本，与英语数据的过程相比只有少量修改：
  - **生成**：在后训练的早期轮次中，我们探索了在 0.2−1 范围内随机选择温度超参数，以实现多样化的生成。高温下，多语言提示的响应可以变得创造性和鼓舞人心，但也容易受到不必要或不自然的代码转换的影响。在后训练的最后轮次中，我们使用恒定的 0.6 值来平衡权衡。此外，我们还使用专门的系统提示来改进响应的格式、结构和总体可读性。
  - **选择**：在基于奖励模型的选择之前，我们实施了特定于多语言的检查，以确保提示和响应之间的高语言匹配率（例如，罗马化的印地语提示不应期望以印地语天城文脚本回应）。

- **翻译数据**：我们尽量避免使用机器翻译的数据来微调模型，以防止翻译腔（Bizzoni 等人，2020 年；Muennighoff 等人，2023 年）或可能的名称偏见（Wang 等人，2022a 年）、性别偏见（Savoldi 等人，2021 年）或文化偏见（Ji 等人，2023 年）。此外，我们旨在防止模型只暴露于植根于英语文化背景的任务中，这可能无法代表我们旨在捕捉的语言和文化多样性。我们对此有一个例外，并将我们的合成定量推理数据（见4.3.3节详情）翻译成非英语语言，以提高非英语语言中的定量推理能力。由于这些数学问题的语言性质简单，翻译样本的质量几乎没有问题。我们观察到，通过添加这些翻译数据，在 MGSM（Shi 等人，2022 年）上取得了强劲的增长。

4.3.3 数学和推理

我们将推理定义为执行多步骤计算并得出正确最终答案的能力。几个挑战指导我们训练在数学推理方面表现出色的模型的方法：

- **缺乏提示**：随着问题的复杂性增加，用于监督式微调（SFT）的有效提示或问题数量减少。这种稀缺性使得为教授模型各种数学技能创建多样化和具有代表性的训练数据集变得困难（Yu 等人，2023 年；Yue 等人，2023 年；Luo 等人，2023 年；Mitra 等人，2024 年；Shao 等人，2024 年；Yue 等人，2024b 年）。

- **缺乏真实思考过程链**：有效的推理需要逐步解决方案以促进推理过程（Wei 等人，2022c 年）。然而，通常缺少真实思考过程链，这对于指导模型如何一步步分解问题并得出最终答案至关重要（Zelikman 等人，2022 年）。

- **中间步骤可能不正确**：当使用模型生成的思考过程链时，中间步骤可能并不总是正确的（Cobbe 等人，2021 年；Uesato 等人，2022 年；Lightman 等人，2023 年；Wang 等人，2023a 年）。这种不准确性可能导致最终答案不正确，需要解决。

- **教授模型使用外部工具**：增强模型以利用外部工具，如代码解释器，允许它们通过交错代码和文本进行推理（Gao 等人，2023 年；Chen 等人，2022 年；Gou 等人，2023 年）。这种能力可以显著提高它们的解决问题能力。

- **训练和推理之间的差异**：训练期间模型的微调方式与推理期间的使用方式之间通常存在差异。在推理期间，微调模型可能与人类或其他模型交互，要求它使用反馈改进其推理。确保训练和真实世界使用之间的一致性对于维持推理性能至关重要。

为了应对这些挑战，我们应用了以下方法：

- **解决提示缺乏问题**：我们从数学背景中获取相关的预训练数据，并将其转换为问答格式，然后可用于监督式微调。此外，我们确定模型表现不佳的数学技能，并积极从人类那里获取提示，以教授模型这些技能。为了促进这个过程，我们创建了一个数学技能分类（Didolkar 等人，2024 年），并要求人类相应地提供相关的提示/问题。
- **用逐步推理痕迹增强训练数据**：我们使用 Llama 3 为一组提示生成逐步解决方案。对于每个提示，模型产生不同数量的生成。然后根据正确答案过滤这些生成（Li 等人，2024a 年）。我们还进行自我验证，使用 Llama 3 验证特定的逐步解决方案是否适用于给定问题。这个过程通过消除模型未产生有效推理痕迹的实例，提高了微调数据的质量。
- **过滤不正确的推理痕迹**：我们训练结果和逐步奖励模型（Lightman 等人，2023 年；Wang 等人，2023a 年）来过滤训练数据，其中中间推理步骤是不正确的。这些奖励模型用于消除具有无效逐步推理的数据，确保微调的高质量数据。对于更具挑战性的提示，我们使用带有学习逐步奖励模型的蒙特卡洛树搜索（MCTS）生成有效的推理痕迹，进一步提高了收集高质量推理数据的能力（Xie 等人，2024 年）。
- **交错代码和文本推理**：我们提示 Llama 3 通过文本推理和相关 Python 代码的组合来解决推理问题（Gou 等人，2023 年）。代码执行被用作反馈信号，以消除推理链无效的情况，确保推理过程的正确性。
- **从反馈和错误中学习**：为了模拟人类反馈，我们利用不正确的生成（即导致不正确推理痕迹的生成）并通过提示 Llama 3 产生正确的生成来进行错误纠正（An 等人，2023b 年；Welleck 等人，2022 年；Madaan 等人，2024a 年）。使用来自不正确尝试的反馈并纠正它们的迭代过程有助于提高模型准确推理并从错误中学习的能力。

4.3.4 长文本

在最后的预训练阶段，我们将 Llama 3 的上下文长度从 8K 令牌扩展到 128K 令牌（有关更多详细信息，请参阅第**??**节）。与预训练类似，我们发现在微调过程中，我们必须仔细调整配方，以平衡短文本和长文本能力。

SFT 和合成数据生成 简单地应用我们现有的仅使用短文本数据的 SFT 配方，导致长文本能力相比预训练出现了显著的退步，这突显了在我们的 SFT 数据混合中纳入长文本数据的必要性。然而，在实践中，由于阅读长文本的乏味和耗时，让人类注释这类示例在很大程度上是不切实际的，因此我们主要依赖合成数据来填补这一空白。我们使用早期版本的 Llama 3 根据关键的长文本用例生成合成数据：（可能是多轮的）问答、长文档的摘要以及对代码库的推理，我们将在下面更详细地描述它们。

- **问答**：我们从预训练混合中精心策划了一系列长文档。我们将这些文档分割成 8K 令牌的块，并提示早期版本的 Llama 3 模型根据随机选择的块生成 QA 对。在训练过程中，整个文档被用作上下文。

- **摘要**：我们通过对长文本文档进行分层摘要来应用长文本摘要，首先使用我们最强的 Llama 3 8K 上下文模型对 8K 输入长度的块进行摘要，然后对摘要进行摘要。在训练中，我们提供完整文档，并提示模型在保留所有重要细节的同时对文档进行摘要。我们还根据文档的摘要生成 QA 对，并用需要对整个长文档全局理解的问题提示模型。

- **长文本代码推理**：我们解析 Python 文件以识别 `import` 语句并确定它们的依赖关系。从这里，我们选择最常依赖的文件，特别是至少被其他五个文件引用的文件。我们从一个代码库中移除其中一个关键文件，并提示模型识别哪些文件依赖于缺失的文件，并生成必要的缺失代码。

我们根据序列长度（16K、32K、64K 和 128K）进一步对这些合成生成的样本进行分类，以便更精细地定位输入长度。

通过仔细的消融研究，我们观察到将 0.1% 的合成生成的长文本数据与原始短文本数据混合，可以在短文本和长文本基准测试中优化性能。

DPO 我们观察到，只要 SFT 模型适合长文本任务，仅使用短文本训练数据在 DPO 中就不会对长文本性能产生负面影响。我们怀疑这是由于我们的 DPO 配方比 SFT 拥有更少的优化步骤。鉴于这一发现，我们在长文本 SFT 检查点上保持 DPO 的标准短文本配方。

4.3.5 工具使用

教会大型语言模型（LLMs）使用工具，如搜索引擎或代码解释器，极大地扩展了它们能够解决的任务范围，将它们从纯聊天模型转变为更通用的助手（Nakano 等人，2021 年；Thoppilan 等人，2022 年；Parisi 等人，2022 年；Gao 等人，2023 年；Mialon 等人，2023a 年；Schick 等人，2024 年）。我们训练 Llama 3 与以下工具进行交互：

- **搜索引擎** Llama 3 被训练使用 Brave Search 来回答问题，这些答案超出了它的知识截止日期，或者需要从网络检索特定信息。

- **Python 解释器** Llama 3 可以生成并执行代码以执行复杂计算、读取用户上传的文件，并根据这些文件解决任务，如问答、摘要、数据分析或可视化。

- **数学计算引擎** Llama 3 可以使用 Wolfram Alpha API 来更准确地解决数学、科学问题，或从 Wolfram 的数据库检索准确信息。

所得到的模型能够在聊天设置中使用这些工具来解决用户的查询，包括在多轮对话中。如果查询需要多次调用工具，模型可以编写逐步计划，按顺序调用工具，并在每次工具调用后进行推理。

我们还提高了 Llama 3 的零样本工具使用能力——在给定上下文中，可能未见过的工具定义和用户查询，我们训练模型生成正确的工具调用。

实现 我们将核心工具实现为具有不同方法的 Python 对象。零样本工具可以作为具有描述、文档（即，如何使用它们的示例）的 Python 函数实现，模型只需要函数的签名和文档字符串作为上下文来生成适当的调用。我们还将函数定义和调用转换为 JSON 格式，例如，用于 Web API 调用。所有工具调用都由 Python 解释器执行，该解释器必须在 Llama 3 系统提示中启用。核心工具可以在系统提示中单独启用或禁用。

数据收集 与 Schick 等人（2024 年）不同，我们依赖人工标注和偏好来教会 Llama 3 使用工具。与通常在 Llama 3 中使用的后训练流程有两个主要区别：

- 对于工具，对话通常包含多个助手消息（例如，调用工具并推理工具输出）。因此，我们在消息级别进行注释以收集细粒度反馈：注释者在两个具有相同上下文的助手消息之间提供偏好，或者，如果两个都包含重大问题，则编辑其中之一。选择或编辑的消息然后被添加到上下文中，对话继续。这为助手调用工具的能力和对工具输出的推理提供了人工反馈。注释者不能对工具输出进行排名或编辑。
- 我们不执行拒绝采样，因为我们没有在我们的工具基准测试中观察到增益。

为了加速注释过程，我们首先通过在以前 Llama 3 检查点生成的合成数据上进行微调，引导基本工具使用能力。因此，注释者需要进行的编辑较少。同样，随着 Llama 3 在开发过程中逐渐改进，我们逐步复杂化我们的人工标注协议：我们从单轮工具使用注释开始，然后转向对话中的工具使用，最后注释多步工具使用和数据分析。

工具数据集 为了为工具使用应用创建数据，我们利用以下程序：

- **单步工具使用**：我们首先通过少量生成的合成用户提示开始，这些提示在构建时就需要调用我们的核心工具之一（例如，超出我们知识截止日期的问题）。然后，仍然依赖少量生成，我们为这些提示生成适当的工具调用，执行它们，并将输出添加到模型的上下文中。最后，我们再次提示模型根据工具输出生成对用户查询的最终答案。我们最终得到以下形式的轨迹：系统提示、用户提示、工具调用、工具输出、最终答案。我们还筛选了约 30% 的数据集，以移除无法执行的工具调用或其他格式问题。
- **多步工具使用**：我们遵循类似的协议，首先生成合成数据以教授模型基本的多步工具使用能力。为此，我们首先提示 Llama 3 生成至少需要两次工具调用的用户提示，这些可以是相同或不同的核心工具。然后，根据这些提示，我们少量提示 Llama 3 生成一个解决方案，包括交错的推理步骤和工具调用，类似于 ReAct（Yao 等人，2022 年）。见图 10，展示了 Llama 3 执行涉及多步工具使用的任务的示例。
- **文件上传**：我们注释以下文件类型：.txt、.docx、.pdf、.pptx、.xlsx、.csv、.tsv、.py、.json、.jsonl、.html、.xml。我们的提示基于提供的文件，并要求摘要文件内容，查找和修复错误，优化代码片段，执行数据分析或可视化。见图 11，展示了 Llama 3 执行涉及文件上传的任务的示例。

在对这个合成数据进行微调后，我们在包括多轮交互、超过三步工具使用，以及工具调用没有得到满意答案的多样化和具有挑战性的场景中收集人工标注。我们通过不同的系统提示增强我们的合成数据，以教会模型仅在激活时使用工具。为了防止模型对简单查询调用工具，我们还添加了来自简单数学或问答数据集（Berant 等人，2013 年；Koncel-Kedziorski 等人，2016 年；Joshi 等人，2017 年；Amini 等人，2019 年）的查询及其响应，但没有工具，但在系统提示中启用了工具。

零样本工具使用数据 我们通过在大量多样的部分合成数据集（函数定义、用户查询、相应的调用）上进行微调，提高了 Llama 3 的零样本工具使用能力（也称为函数调用）。我们在一组未见过的工具上评估我们的模型。

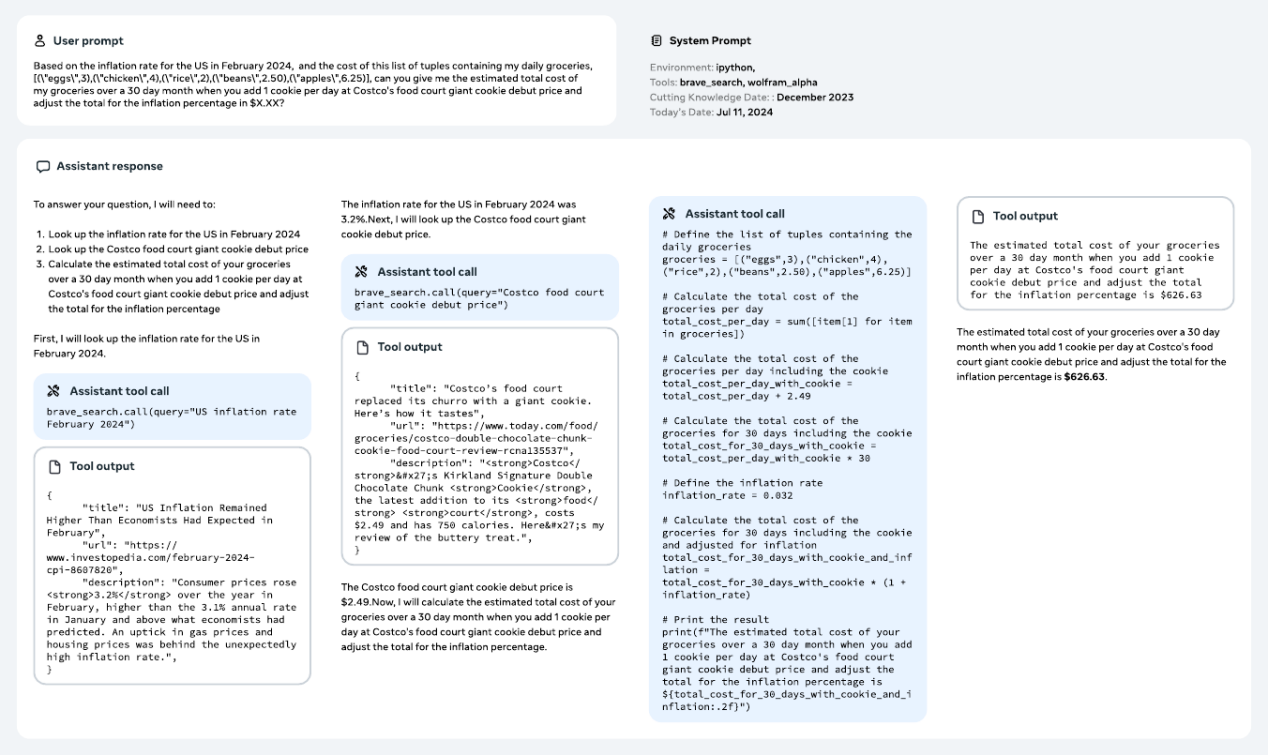

图 17: 多步工具使用。Llama 3 执行多步规划、推理和调用工具来解决任务的示例。

- **单个、嵌套和并行函数调用**：调用可以是简单的，嵌套的，即我们把一个函数调用作为另一个函数的参数传递，或者是并行的，即模型返回一系列独立的函数调用。生成多样化的函数、查询和真实情况可能具有挑战性（Mekala 等人，2024 年），我们依靠挖掘 Stack（Kocetkov 等人，2022 年）来使我们的合成用户查询基于真实函数。更具体地说，我们提取函数调用及其定义，进行清理和过滤，例如缺少文档字符串或不可执行的函数，并使用 Llama 3 生成与函数调用相对应的自然语言查询。

- **多轮函数调用**：我们还为带有函数调用的多轮对话生成合成数据，遵循 Li 等人（2023b 年）提出的协议。我们使用多个代理生成领域、API、用户查询、API 调用和响应，同时确保生成的数据涵盖多样化的领域和现实的 API。所有代理都是以不同方式提示的 Llama 3 的变体，根据它们的角色进行协作，并以逐步的方式进行协作。

4.3.6 事实性

幻觉仍然是大型语言模型面临的一个主要挑战。模型往往过于自信，即使在它们知之甚少的领域也是如此。尽管存在这些缺陷，它们通常仍被用作知识库，这可能导致诸如错误信息传播等危险结果。虽然我们认识到事实性可以超越幻觉，但在这里我们采取了以幻觉为先的方法。我们遵循的原则是，训练后应该使模型“知道它所知道的”，而不是增加知识（Gekhman 等人，2024 年；Mielke 等人，2020 年）。我们的主要方法包括生成数据，使模型生成与预训练数据中呈现的事实数据子集一致。为了实现这一点，我们开发了一种利用 Llama 3 上下文能力的知识探测技术。这个数据生成过程包括以下步骤：

1. 从预训练数据中提取数据片段。

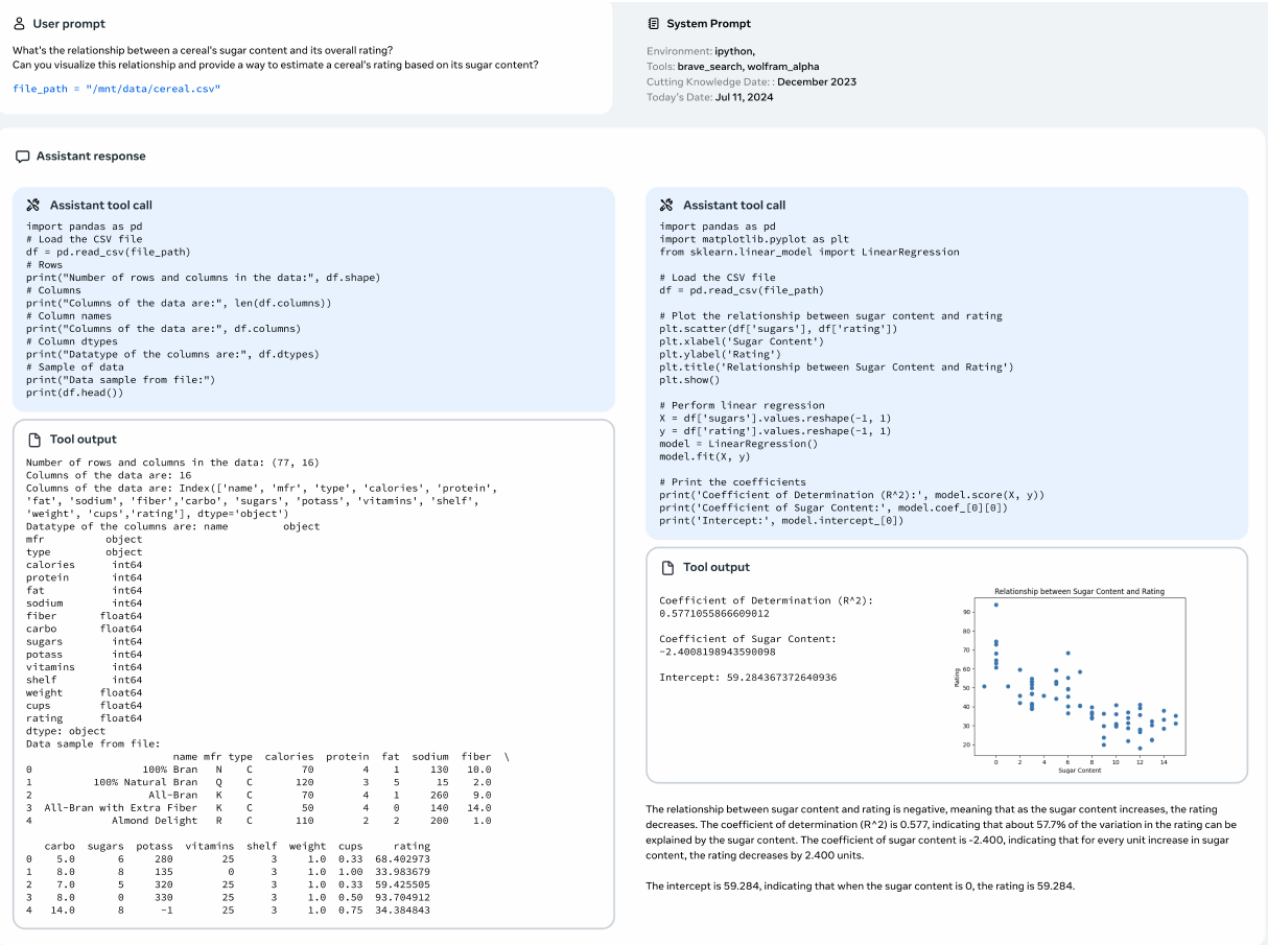

图 18: 处理文件上传。Llama 3 对上传文件进行分析和可视化的示例。

2. 通过提示 Llama 3，基于这些片段（上下文）生成一个事实性问题。
3. 从 Llama 3 对问题进行采样响应。
4. 使用原始上下文作为参考，以 Llama 3 为裁判，对生成的正确性进行评分。
5. 使用 Llama 3 为裁判，对生成的信息量进行评分。
6. 对于在生成中一贯信息量大但不正确的响应，使用 Llama 3 生成拒绝回答。

我们使用从知识探测生成的数据，鼓励模型只回答它所知道的问题，并拒绝回答那些它不确定的问题。此外，预训练数据并不总是事实一致或正确的。因此，我们还收集了有限的标记事实性数据集，这些数据涉及存在事实矛盾或错误陈述的敏感话题。

4.3.7 可引导性

可引导性是将模型的行动和结果引导至满足开发者和用户规范的能力。由于 Llama 3 是一个通用的基础模型，它应该能够轻松地被引导至不同的下游用例。对于 Llama 3，我们专注于通过自然语言指令的系统提示来增强其可引导性，尤其是在响应长度、格式、语调和角色/个性方面。

数据收集 我们通过要求注释者为 Llama 3 设计不同的系统提示，在普通英语类别中收集可引导性偏好样本。然后，注释者与模型进行对话，评估它们在对话过程中遵循系统提示中定义的指令的一致性。我们在下面展示了一个用于增强可引导性的定制系统提示示例：

> You are a helpful and cheerful AI Chatbot that acts as a meal plan assistant for busy families. The family consists of 2 adults, 3 teenagers, and 2 preschoolers. Plan two or three days at a time and use leftovers or extra ingredients for the second day's plan. The user will let you know if they want two or three days. If they don't, assume three days. Each plan should include breakfast, lunch, snack, and dinner. Ask the user if they approve of the plan or need adjustments. After they approve provide a grocery list with family size in mind. Always keep family preferences in mind and if there's something that they don't like provide a substitution. If the user is not feeling inspired then ask them what's the one place they wish they could visit on vacation this week and then suggest meals based on that location's culture. Weekend meals can be more complex. Weekday meals should be quick and easy. For breakfast and lunch, easy food like cereal, English muffins with pre-cooked bacon, and other quick easy foods are preferred. The family is busy. Be sure to ask if they have essentials and favorites on hand like coffee or energy drinks so they don't forget to buy it. Remember to be budget-conscious unless it's a special occasion.

建模 在我们收集了偏好数据后，我们利用这些数据在奖励建模、拒绝采样、SFT（监督式微调）和 DPO（直接偏好优化）中，以增强 Llama 3 的可引导性。

5 Results

此处只有概述，具体请参考原文。

我们对 Llama 3 进行了一系列广泛的评估，调查了以下方面的性能：（1）预训练语言模型，（2）后训练语言模型，以及（3）Llama 3 的安全特性。我们在下面的单独小节中展示了这些评估的结果。

5.1 Pre-trained Language Model 预训练语言模型

在本节中，我们报告了我们预训练的 Llama 3（第 3 节）的评估结果，并与各种其他尺寸相当的模型进行了比较。我们尽可能重现了竞争对手模型的结果。对于非 Llama 模型，我们报告了公开报告或（如果可能）我们自己重现的结果中的最佳分数。这些评估的具体细节，包括诸如样本数量、指标以及其他相关超参数和设置等配置，可以在我们的 Github 仓库中访问。此外，我们还发布了作为评估一部分生成的数据，这些数据可以在 Huggingface 上找到。我们在标准基准（5.1.1 节）上评估我们模型的质量，对于多选题设置变化的鲁棒性（5.1.2 节），以及对抗性评估（5.1.3 节）。我们还进行了污染分析，以估计我们的评估受到训练数据污染影响的程度（5.1.4 节）。

5.2 Post-trained Language Model 后训练语言模型

我们展示了我们的 Llama 3 后训练模型在不同能力基准测试上的结果。与预训练类似，我们发布了作为评估一部分生成的数据，这些数据可以在 Huggingface 上公开获取。有关我们评估设置的更多详细信息可以在这里找到。

基准测试和指标。表 16 概述了所有基准测试，按能力组织。我们通过与每个基准测试的提示进行精确匹配来对后训练数据进行去污染处理。除了标准的学术基准测试外，我们还对不同能力进行了广泛的人工评估。详细信息在第 5.3 节中提供。

实验设置。我们采用与预训练阶段类似的实验设置，并对 Llama 3 进行比较分析，同时与其他尺寸和能力相当的模型进行对比。在可能的情况下，我们自行评估其他模型的性能，并将结果与报告的数据进行比较，选择最佳分数。您可以在此处找到有关我们评估设置的更多详细信息。

Category	Benchmarks
General	MMLU (**?**), MMLU-Pro (**?**), IFEval (**?**)
Math and reasoning	GSM8K (**?**), MATH (**?**), GPQA (**?**), ARC-Challenge (**?**)
Code	HumanEval (**?**), MBPP (**?**), HumanEval+ (**?**), MBPP EvalPlus (base) (**?**), MultiPL-E (**?**)
Multilinguality	MGSM (**?**), Multilingual MMLU (internal benchmark)
Tool-use	Nexus (**?**), API-Bank (**?**), API-Bench (**?**), BFCL (**?**)
Long context	ZeroSCROLLS (**?**), Needle-in-a-Haystack (**?**), InfiniteBench (**?**)

表 1: Post-training benchmarks by category. Overview of all benchmarks we use to evaluate post-trained Llama 3 models, ordered by capability.

5.3 Human Evaluations 人类评估

除了在标准基准集上的评估外，我们还进行了一系列人类评估。这些评估使我们能够测量和优化模型性能的更细微方面，例如我们模型的语气、冗长程度和对细微差别和文化背景的理解。表 22 展示了零样本工具使用基准测试。我们报告了在 Nexus（Srinivasan 等人，2023 年）、API-Bank（Li 等人，2023b）、API-man 评估（Patil 等人，2023 年）和 BFCL（Yan 等人，2024 年）中的函数调用准确性。这些评估紧密反映了用户体验，提供了模型在现实场景中表现的见解。

提示收集 我们收集了涵盖广泛类别和难度的高质量提示。为此，我们首先开发了一个分类法，包括类别和子类别，以尽可能多地捕捉模型的能力。我们利用这一分类法收集了约 7,000 个提示，涵盖六种独立能力（英语、推理、编程、印地语、西班牙语和葡萄牙语）和三种多轮能力（英语、推理和编程）。我们确保在每个类别中，提示在子类别中均匀分布。我们还根据难度将每个提示分为三个级别，并确保我们的提示集合包含大约 10% 简单提示、30% 中等提示和 60% 困难提示。所有人类评估提示集都经过了严格的质量保证流程。建模团队无法访问我们的人类评估提示，以防止测试集的意外污染或过拟合。

评估过程 为了进行两个模型的成对人类评估，我们询问人类标注者他们更喜欢两个模型响应中的哪一个（由不同模型产生）。标注者使用 7 点量表进行评分，使他们能够表明一个模型响应是否远优于、优于、略优于或与另一个模型响应大致相同。当标注者表示一个模型响应优于或远优于另一个模型响应时，我们认为这是该模型的“胜利”。我们在模型之间进行成对比较，报告提示集中每项能力的胜率。

结果 我们利用人类评估流程来比较 Llama 3 405B 与 GPT-4（0125 API 版本）、GPT-40（API 版本）以及 Claude 3.5 Sonnet（API 版本）。这些评估的结果如图 17 所示。我们观察到，Llama 3405B 的表现与 GPT-4 的 0125 API 版本大致相当，而在与 GPT-40 和 Claude 3.5 Sonnet 的比较中则呈现出混合结果（有胜有负）。在几乎所有能力上，Llama 3 和 GPT-4 的胜率都在误差范围内。在多轮推理和编码任务上，Llama 3405B 优于 GPT-4，但在多语言（印地语、西班牙语和葡萄牙语）提示上表现不佳。Llama 3 在英语提示上与 GPT-40 相当，在多语言提示上与 Claude 3.5 Sonnet 相当，并在单轮和多轮英语提示上优于 Claude 3.5 Sonnet。然而，在编码和推理等能力上，它落后于 Claude 3.5 Sonnet。从定性角度看，我们发现模型在人类评估中的表现深受模型语调、响应结构和冗长程度等细微因素的影响——这些因素是我们训练后优化过程中的重点。总体而言，我们的人类评估结果与标准基准评估结果一致：Llama 3405B 与行业领先模型非常具有竞争力，使其成为表现最佳的公开可用模型。

局限性 所有人类评估结果都经过了严格的数据质量保证流程。然而，由于定义模型响应评估的客观标准具有挑战性，人类评估仍可能受到人类标注者的个人偏见、背景和偏好影响，这可能导致结果不一致或不可靠。

5.4 Safety 安全性

我们专注于评估 Llama 3 在安全且负责任的方式下生成内容的能力，同时最大化有用信息。我们的安全工作始于预训练阶段，主要形式为:

数据清洗和过滤 然后我们描述了我们的安全微调方法，重点是如何训练模型以符合特定的安全政策，同时仍然保持有用性。我们分析了 Llama 3 的每项能力，包括多语言、长上下文、工具使用和各种多模态能力，以衡量我们安全缓解措施的有效性。

随后，我们描述了对网络安全和化学及生物武器风险提升的评估。提升指的是与使用现有可用技术（如网络搜索）相比，新技术发展引入的额外风险。

然后我们描述了如何利用红队测试来迭代识别和应对跨能力的安全风险，并进行残余风险评估。

最后，我们描述了系统级安全，即围绕模型本身输入和输出的分类器的发展和编排，以进一步增强安全性，并使开发者更容易为各种用例定制安全措施，并以更负责任的方式部署生成式 AI。

6 Inference

我们研究了两种主要技术来使 Llama 3 405B 模型的推理过程高效：（1）流水线并行性；（2）FP8 量化。我们已经公开发布了我们的 FP8 量化实现。

6.1 流水线并行性

当使用 BF16 数字表示模型参数时，Llama 3 405B 不适合单台机器上的 8 个 Nvidia H100 GPU 的 GPU 内存。为了解决这个问题，我们在两台机器上的 16 个 GPU 上使用 BF16 精度进行模型推理并行化。在每台机器内，高 NVLink 带宽使得可以使用张量并行性（Shoeybi 等人，2019）。然而，在节点之间，连接带宽较低且延迟较高，因此我们使用流水线并行性（Huang 等人，2019）。

在使用流水线并行性进行训练时，气泡是主要的效率问题（见第 3.3 节）。然而，在推理期间它们不是问题，因为推理不涉及需要流水线冲洗的反向传递。因此，我们使用微批处理来提高流水线并行推理的吞吐量。

我们评估了在推理工作负载中使用两个微批处理的影响，输入令牌为 4,096，输出令牌为 256，分别在推理的键值缓存预填充阶段和解码阶段。我们发现微批处理提高了相同本地批量大小的推理吞吐量；见图 24。这些改进来自于微批处理在这两个阶段使微批处理能够并发执行。

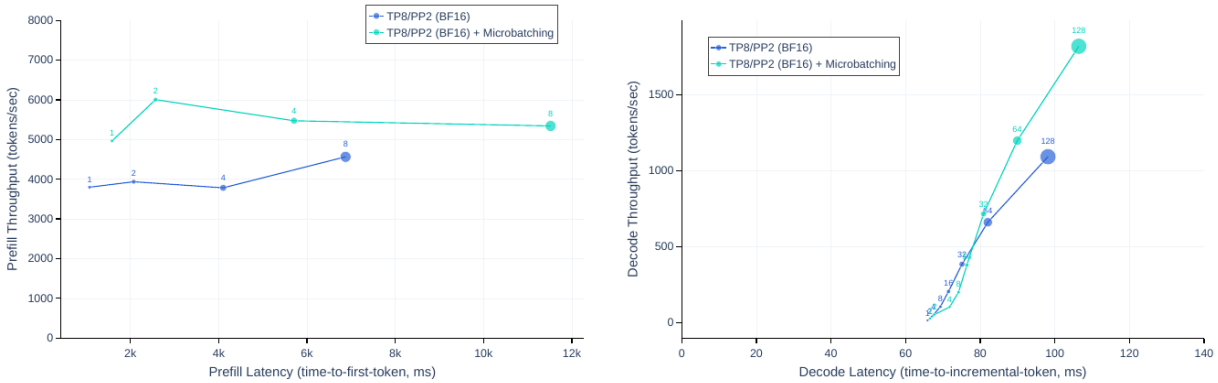

图 19: 图 24 微批处理对推理吞吐量和延迟的影响，左侧：预填充阶段，右侧：解码阶段。图中的数字对应于（微）批量大小。

由于微批处理而增加的额外同步点也增加了延迟，但总体上，微批处理仍然导致了更好的吞吐量-延迟权衡。

6.2 FP8 量化

我们利用 H100 GPU 对 FP8 的原生支持进行低精度推理实验。为了实现低精度推理，我们对模型内的大多数矩阵乘法应用 FP8 量化。特别是，我们量化了模型中前馈网络层中的大多数参数和激活，这些层大约占推理计算时间的 50

- 1. 与 Zhang 等人（2021）类似，我们不在第一层和最后一层 Transformer 层进行量化。
- 2. 高困惑度的令牌，如日期，可能导致大的激活值。这可能导致 FP8 中的高动态缩放因子和不可忽略数量的下溢，导致解码错误。为了解决这个问题，我们将动态缩放因子上限定为 1200。
- 3. 我们使用按行量化，为参数和激活矩阵计算跨行的缩放因子（见图 25）。我们发现这种方法比按张量量化的方法更有效。

6.2.1 量化误差的影响

标准基准测试通常表明，即使没有这些缓解措施，FP8 推理的性能也与 BF16 推理相当。然而，我们发现这样的基准测试并没有充分反映 FP8 量化的影响。当缩放因子没有上界时，模型偶尔会产生损坏的响应，尽管基准性能很强。我们发现，与其依赖基准测试来衡量量化引起的分布变化，不如分析使用 FP8 和 BF16 产生的 100,000 个响应的奖励模型分数的分布。图 26 显示了我们的量化方法的奖励分布结果。图中的结果显示，我们的 FP8 量化方法对模型的响应影响非常有限。

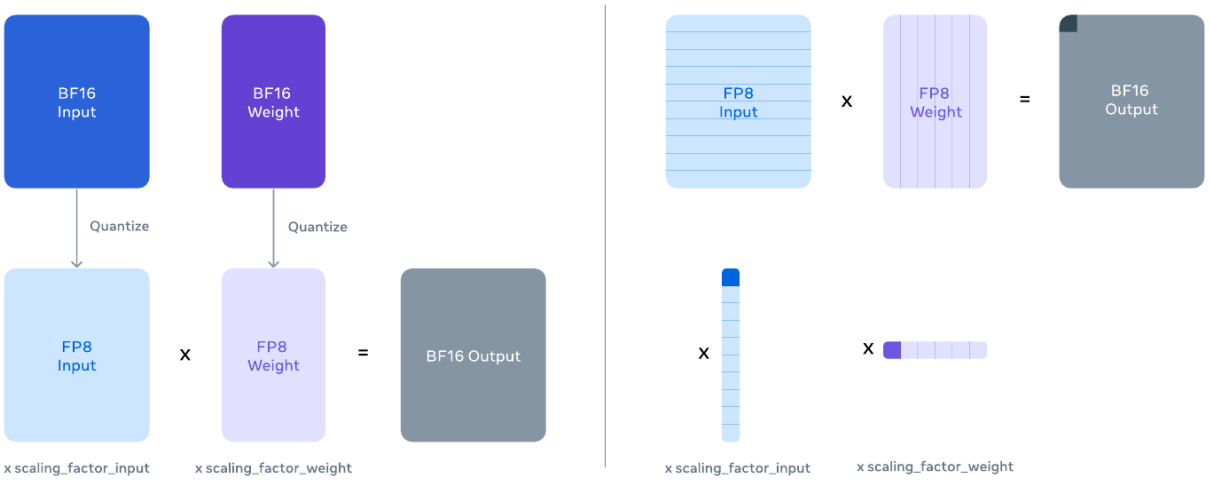

图 20: 图 25 展示了张量级和按行 FP8 量化的示意图。右侧：按行量化允许使用比左侧：张量级量化更细粒度的激活因子。

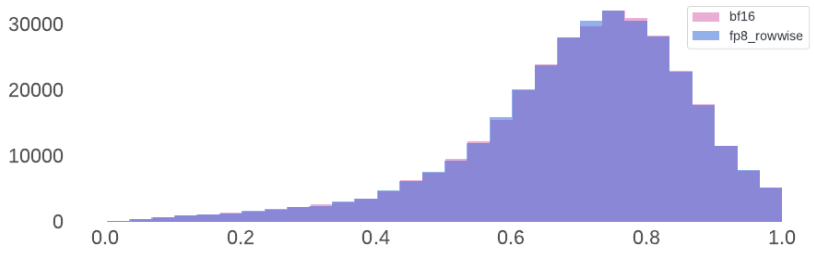

图 21: 图 26 Llama 3 405B 使用 BF16 和 FP8 推理的奖励分数分布。我们的 FP8 量化方法对模型的响应几乎没有影响。

6.2.2 效率的实验评估

图 27 描绘了在预填充和解码阶段使用 Llama 3 405B 进行 FP8 推理的吞吐量-延迟权衡，输入令牌为 4,096，输出令牌为 256。该图比较了第 6.1 节中描述的两台机器 BF16 推理方法的效率。结果表明，使用 FP8 推理在预填充阶段的吞吐量提高了高达 50

7 Vision Experiments

我们进行了一系列实验，通过一种由两个主要阶段组成的组合方法将视觉识别能力整合到 Llama 3 中。首先，我们通过在两个模型之间引入和训练一组交叉注意力层（Alayrac 等人，2022），将预训练的图像编码器（Xu 等人，2023）和预训练的语言模型组合起来，用于大量图像-文本对。这导致形成了图23所

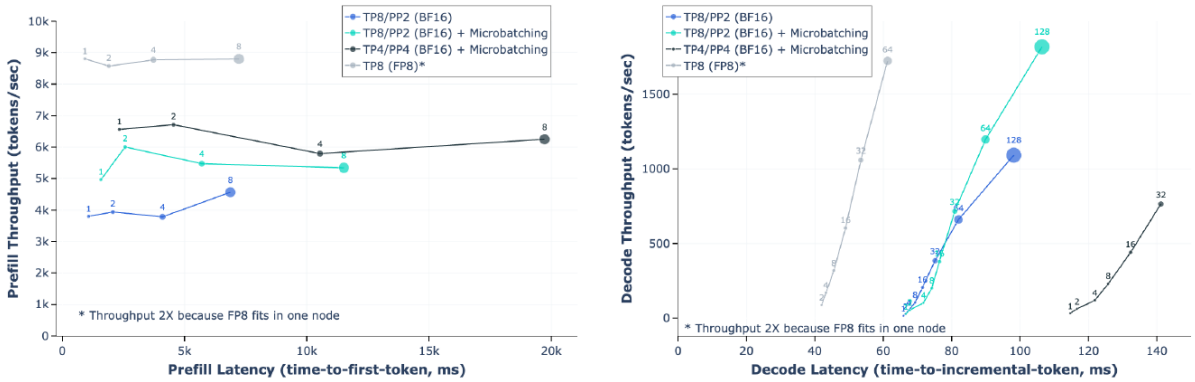

图 22: 图 27 使用 Llama 3 405B 进行 FP8 推理与使用不同流水线并行设置的 BF16 推理的吞吐量-延迟权衡。左侧：预填充的结果。右侧：解码的结果。

示的模型。其次，我们引入了时间聚合层和额外的视频交叉注意力层，这些层在大量视频-文本对上操作，以学习模型识别和处理视频中的时间信息。

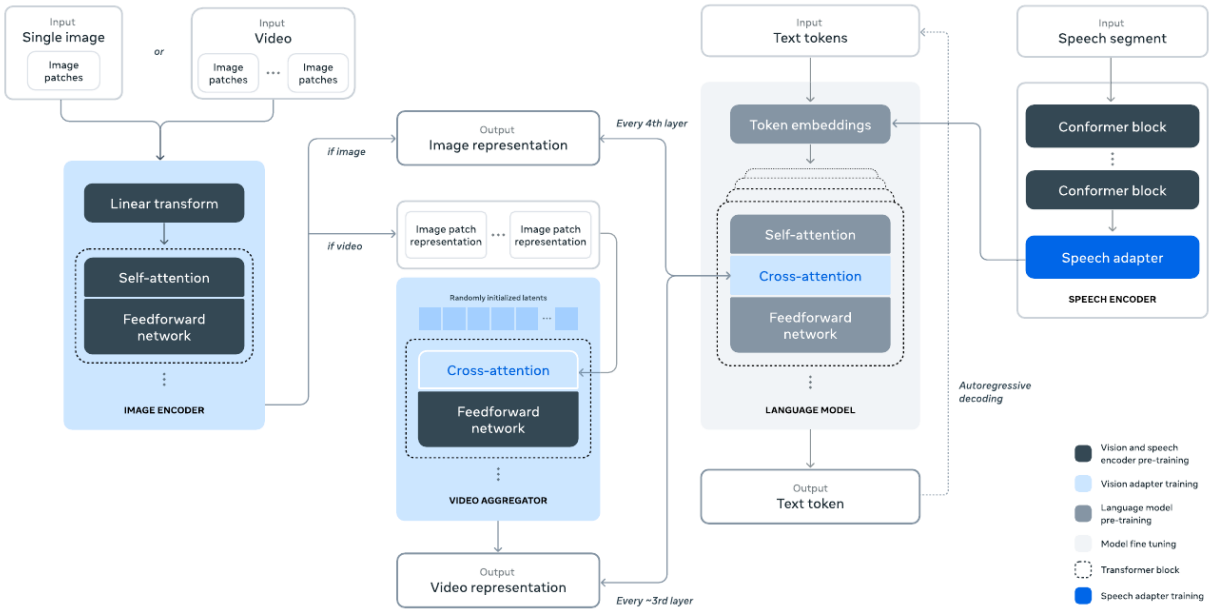

图 23: 本文研究的向 Llama 3 添加多模态能力的组合方法的示意图。这种方法导致了一个经过五个阶段训练的多模态模型：(1) 语言模型预训练，(2) 多模态编码器预训练，(3) 视觉适配器训练，(4) 模型微调，以及 (5) 语音适配器训练。

采用组合方法进行基础模型开发有几个优点：（1）它使我们能够并行开发视觉和语言建模能力；（2）它避免了视觉和语言数据联合预训练的复杂性，这些复杂性源于视觉数据的标记化、来自不同模态的标记的背景困惑度差异以及模态之间的竞争；（3）它保证了模型在仅文本任务上的性能不会因引入视觉识别能力而受到影响，并且（4）交叉注意力架构确保我们不需要通过越来越多的 LLM 骨干（特别是每个 transformer 层中的前馈网络）传递全分辨率图像，使推理过程更加高效。我们注意到我们的多模态模型仍在开发中，尚未准备好发布。

在第 7.6 节和 7.7 节中介绍我们的实验结果之前，我们描述了我们用于训练视觉识别能力的数据，视觉组件的模型架构，我们如何扩展这些组件的训练，以及我们的预训练和后训练配方。

7.1 数据

我们分别描述了我们的图像和视频数据。

7.1.1 图像数据

我们的图像编码器和适配器在图像-文本对上进行训练。我们通过一个复杂的数据处理流程构建这个数据集，该流程包括四个主要阶段：（1）质量过滤，（2）感知去重，（3）重新采样，和（4）光学字符识别。我们还应用了一系列安全缓解措施。

- **质量过滤**我们实施了质量过滤器，通过（Radford 等人，2021）产生的低对齐分数等启发式方法去除非英语字幕和低质量字幕。具体来说，我们删除了所有低于某个 CLIP 分数的图像-文本对。

- **去重**对大规模训练数据集进行去重有助于模型性能，因为它减少了在冗余数据上花费的训练计算（Esser 等人，2024；Lee 等人，2021；Abbas 等人，2023）和记忆（Carlini 等人，2023；Somepalli 等人，2023）。因此，出于效率和隐私原因，我们对训练数据进行了去重。为此，我们使用最先进的 SSCD 复制检测模型（Pizzi 等人，2022）的内部版本在规模上去除图像重复。对于所有图像，我们首先使用 SSCD 模型计算 512 维表示。我们使用这些嵌入在数据集中的所有图像中为每张图像执行最近邻（NN）搜索，使用余弦相似性度量。我们将相似度超过某个阈值的示例定义为重复项。我们使用连接组件算法对这些重复项进行分组，并在每个连接组件中仅保留一个图像-文本对。我们通过以下方式提高去重流程的效率：（1）使用 k-means 聚类预聚类数据，以及（2）使用 FAISS（Johnson 等人，2019）进行 NN 搜索和聚类。

- **重新采样**我们通过类似于 Xu 等人（2023）；Mahajan 等人（2018）；Mikolov 等人（2013）的重新采样确保图像-文本对的多样性。首先，我们通过解析高质量文本源构建 n-gram 词汇表。接下来，我们计算数据集中每个词汇 n-gram 的频率。然后我们按以下方式重新采样数据：如果标题中的任何 n-gram 在词汇表中出现次数少于 T 次，我们保留相应的图像-文本对。否则，我们以概率 pT/f_i 独立地采样标题中的每个 n-gram n_i，其中 f_i 表示 n-gram n_i 的频率；如果任何 n-gram 被采样，我们保留图像-文本对。这种重新采样有助于在低频类别和细粒度识别任务上的性能。

- **光学字符识别**我们通过提取图像中书写的文本并将其与字幕连接起来，进一步改善我们的图像-文本数据。使用专有的光学字符识别（OCR）流程提取书面文本。我们观察到，将 OCR 数据添加到训练数据中大大改善了需要 OCR 能力的任务，例如文档理解。

转录文档 为了提高我们的模型在文档理解任务上的性能，我们将文档页面渲染为图像，并将图像与其各自的文本配对。文档文本直接从源中获取，或通过文档解析流程获取。

安全 我们主要关注确保图像识别的预训练数据集不包含不安全内容，例如性虐待材料（CSAM）（Thiel，2023）。我们使用感知散列方法，如 PhotoDNA（Farid，2021）以及内部专有分类器扫描所有训练图像中的 CSAM。我们还使用专有的媒体风险检索流程来识别和删除我们认为不适合工作场所的图像-文本对，例如，因为它们包含性或暴力内容。我们认为，最小化这类材料在训练数据集中的流行度可以提高最终模型的安全性，而不影响其有用性。最后，我们对训练集中的所有图像进行面部模糊处理。我们使用人类生成的提示测试模型，这些提示引用了附加的图像。

退火数据 我们通过使用 n-gram 重新采样图像-字幕对到较小体积的约 3.5 亿个示例来创建退火数据集。由于 n-gram 重新采样偏好更丰富的文本描述，这选择了更高质量的数据子集。我们用来自五个额外来源的约 1.5 亿个示例增强了生成的数据：

- **视觉定位**我们将文本中的名词短语链接到图像中的边界框或掩码。定位信息（边界框和掩码）以两种方式在图像-文本对中指定。（1）我们在图像上叠加框或掩码，并使用文本中的标记作为参考，类似于标记集（Yang 等人，2023a）。（2）我们直接将标准化的（xmin，ymin，xmax，ymax）坐标插入到文本中，用特殊标记分隔。
- **屏幕截图解析**我们从 HTML 代码渲染屏幕截图，并让模型预测产生屏幕截图中特定元素的代码，类似于 Lee 等人（2023）。感兴趣的元素通过屏幕截图中的边界框指示。
- **问题-答案对**我们包括问题-答案对，使我们能够使用太大而无法在模型微调中使用的问题-答案数据量。
- **合成字幕**我们包括带有由模型的早期版本生成的合成字幕的图像。与原始字幕相比，我们发现合成字幕比原始字幕提供了更全面的图像描述。
- **合成生成的结构化图像**我们还包括了各种领域（如图表、表格、流程图、数学方程和文本数据）的合成生成图像。这些图像伴随着结构化表示，如相应的 markdown 或 LaTeX 符号。

除了提高模型对这些领域的识别能力外，我们发现这些数据对于通过文本模型生成问题-答案对进行微调也很有用。

7.1.2 视频数据

对于视频预训练，我们使用了大量的视频-文本对数据集。我们的数据集是通过多阶段流程精心策划的。我们使用基于规则的启发式方法过滤和清理相关文本，例如确保有最小长度和修正大写。然后，我们运行语言识别模型来过滤掉非英语文本。我们运行 OCR 检测模型来过滤掉叠加文本过多的视频。为确保视频-文本对之间合理的对齐，我们使用 CLIP（Radford 等人，2021）风格的图像-文本和视频-文本对比模型。我们首先使用视频中的单一帧计算图像-文本相似性，并过滤掉相似性低的对，然后随后过滤掉视频-文本对齐度低的对。我们的一些数据包含静态或低运动视频；我们使用基于运动得分的过滤（Girdhar 等人，2023）过滤掉此类数据。我们没有对视频的视觉质量应用任何过滤器，如审美分数或分辨率过滤。

我们的数据集包含平均时长为 21 秒的视频，中位数时长为 16 秒，超过 99% 的视频时长在一分钟内。空间分辨率在 320p 和 4K 视频之间有显著变化，超过 70% 的视频短边大于 720 像素。视频具有不同的长宽比，几乎所有视频的长宽比在 1:2 到 2:1 之间，中位数为 1:1。

7.2 模型架构

我们的视觉识别模型由三个主要部分组成：（1）图像编码器，（2）图像适配器，以及（3）视频适配器。

图像编码器 我们的图像编码器是一个标准的视觉 transformer（ViT; Dosovitskiy 等人，2020），它被训练用于对齐图像和文本（Xu 等人，2023）。我们使用的是 ViT-H/14 变体的图像编码器，它有 6.3 亿参数，这些参数是在 25 亿图像-文本对上训练了五个周期。图像编码器是在 224×224 分辨率的图像上预训练的；图像被分割成 16×16 个相等大小的块（即，每个块的大小为 14×14 像素）。正如先前的工作（如 ViP-Llava（Cai 等人，2024））所展示的，我们观察到通过对比文本对齐目标训练的图像编码器无法保留细粒度的定位信息。为了缓解这个问题，我们采用了多层特征提取，除了最后一层的特征外，还提供了来自第 4 层、第 8 层、第 16 层、第 24 层和第 31 层的特征。

此外，我们在交叉注意力层的预训练之前进一步插入了 8 个门控自注意力层（总共 40 个 transformer 块），以学习对齐特定特征。因此，图像编码器最终总共有 8500 万个参数，加上额外的层。有了多层特征，图像编码器为每个产生的 16×16=256 个块生成了一个 7680 维的表示。在后续的训练阶段，我们没有冻结图像编码器的参数，因为我们发现这可以提高性能，特别是在文本识别等领域。

图像适配器 我们在图像编码器产生的视觉上的标记表示和语言模型产生的标记表示之间引入了交叉注意力层（Alayrac 等人，2022）。交叉注意力层在核心语言模型的每第四个自注意力层之后应用。像语言模型本身一样，交叉注意力层使用广义查询注意力（GQA）以提高效率。交叉注意力层向模型引入了大量的额外可训练参数：对于 Llama 3 405B，交叉注意力层有大约 1000 亿参数。我们分两个阶段预训练我们的图像适配器：（1）初始预训练，然后是（2）退火：

- **初始预训练**我们在上述描述的约 60 亿图像-文本对数据集上预训练我们的图像适配器。出于计算效率的考虑，我们将所有图像调整大小以适应最多四个 336×336 像素的瓦片，我们将瓦片排列以支持不同的长宽比，例如 672×672、672×336 和 1344×336。
- **退火**我们继续在上述描述的退火数据集的约 5 亿图像上训练图像适配器。在退火过程中，我们增加了每个瓦片的图像分辨率，以提高需要更高分辨率图像的任务的性能，例如信息图表的理解。

视频适配器 我们的模型接受最多 64 帧（从完整视频中均匀采样），每帧都由图像编码器处理。我们通过两个组件对视频中的时间结构进行建模：（i）编码后的视频帧通过时间聚合器进行聚合，该聚合器将 32 连续帧合并为一个，（ii）在每第四个图像交叉注意力层之前添加额外的视频交叉注意力层。时间聚合器实现为感知器重采样器（Jaegle 等人，2021；Alayrac 等人，2022）。我们使用每视频 16 帧（聚合为 1 帧）进行预训练，但在监督微调期间将输入帧数增加到 64。对于 Llama 3 7B 和 70B，视频聚合器和交叉注意力层分别有 6 亿和 46 亿参数。

7.3 模型扩展

在将视觉识别组件添加到 Llama 3 之后，模型包含了自注意力层、交叉注意力层和 ViT 图像编码器。为了训练较小的 8B 和 70B 参数模型的适配器，我们发现数据和张量并行的组合是最高效的。在这些规模上，模型或流水线并行性不会提高效率，因为模型参数的聚合将主导计算。然而，在训练 405B 参数模型的适配器时，我们确实使用了流水线并行性（除了数据和张量并行）。在这个规模上的训练引入了除了第 3.3 节中概述的之外的三个新挑战：模型异构性、数据异构性和数值不稳定性。

模型异构性 模型计算是异构的，因为一些 token 比其他 token 执行了更多的计算。特别是，图像 token 由图像编码器和交叉注意力层处理，而文本 token 仅由语言骨干处理。这种异构性导致了流水线并行性调度中的瓶颈。我们通过确保每个流水线阶段包含五层来解决这个问题：即语言骨干中的四个自注意力层和一个交叉注意力层。（回想一下，我们在每第四个自注意力层之后引入一个交叉注意力层。）此外，我们在所有流水线阶段复制图像编码器。因为我们在成对的图像-文本数据上训练，这使我们能够在计算的图像和文本部分之间进行负载平衡。

数据异构性 数据是异构的，因为平均来说，图像比相关文本有更多的 token：一张图像有 2308 个 token，而相关文本平均只有 192 个 token。因此，交叉注意力层的计算比自注意力层的计算需要更多的时间和内存。我们通过在图像编码器中引入序列并行性来解决这个问题，以便每个 GPU 处理大致相同数量的 token。由于平均文本大小相对较短，我们还使用了更大的微批量大小（8 而不是 1）。

数值不稳定性 在将图像编码器添加到模型后，我们发现在 bf16 中执行梯度累积导致了数值不稳定性。这最可能的解释是，图像 token 通过所有交叉注意力层引入到语言骨干中。这意味着图像 token 的表示中的数值偏差对整体计算有巨大的影响，因为错误是累积的。我们通过在 FP32 中执行梯度累积来解决这个问题。

7.4 预训练

图像 我们从预训练的文本模型和视觉编码器权重开始初始化。视觉编码器是解冻的，而文本模型权重则如上所述保持冻结。首先，我们使用每个图像调整大小以适应四个 336×336 像素瓦片的 60 亿图像-文本对来训练模型。我们使用 16384 的全局批量大小和余弦学习率计划，初始学习率为 10×10^{-4}，权重衰减为 0.01。初始学习率是基于小规模实验确定的。然而，这些发现并没有很好地推广到非常长的训练计划，在训练期间损失值停滞不前时，我们几次降低了学习率。在基础预训练之后，我们进一步提高图像分辨率，并在退火数据集上继续使用相同的权重进行训练。优化器通过预热重新初始化为学习率 2×10^{-5}，然后再次遵循余弦计划。

视频 对于视频预训练，我们从上述描述的图像预训练和退火权重开始。我们添加了视频聚合器和交叉注意力层，如架构中所述，随机初始化。我们冻结了模型中的所有参数，除了视频特定的参数（聚合器和视频交叉注意力），并在视频预训练数据上训练它们。我们使用与图像退火阶段相同的训练超参数，学习率有小的差异。我们从完整视频中均匀采样 16 帧，并使用每个 448×448 像素的四个块来表示每一帧。我们在视频聚合器中使用 16 的聚合因子，因此获得一个有效的帧，文本 token 交叉关注它。我们使用 4096 的全局批量大小，190 个 token 的序列长度，在训练期间使用 10^{-4} 的学习率。

7.5 后训练

在本节中，我们描述了我们视觉适配器的后训练配方。预训练后，我们在高度策划的多模态对话数据上对模型进行微调，以启用聊天功能。我们进一步实施直接偏好优化（DPO）以提高人类评估性能，并采用拒绝采样以提高多模态推理能力。最后，我们增加了一个质量调整阶段，我们在非常小的一组高质量对话数据上继续微调模型，这在保留基准测试性能的同时进一步提高了人类评估。

7.5.1 监督微调数据

我们分别描述了图像和视频能力的监督微调（SFT）数据。

图像 我们利用不同的数据集混合进行监督微调。

- 学术数据集。我们使用模板或通过 LLM 重写将高度筛选的现有学术数据集转换为问答对。LLM 重写的目的是用不同的指令增强数据，并提高答案的语言质量。
- 人类注释。我们通过人类注释者收集多模态对话数据，涵盖广泛的任务（开放式问答、字幕、实际用例等）和领域（例如，自然图像和结构化图像）。注释者被提供图像，并被要求编写对话。为确保多样性，我们在不同集群中均匀地聚类大规模数据集和采样图像。此外，我们通过 k 最近邻扩展种子来获取一些特定领域的额外图像。注释者还被提供现有模型的中间检查点，以促进模型在循环中的注释风格，使模型生成可以作为注释者提供的额外人类编辑的起点。这是一个迭代过程，其中模型检查点将定期更新为在最新数据上训练的更好执行版本的模型。这增加了人类注释的体积和效率，同时也提高了它们的质量。
- 合成数据。我们探索使用图像的文本表示和文本输入 LLM 生成合成多模态数据的不同方法。高层思路是利用文本输入 LLM 的推理能力在文本领域生成问答对，并将文本表示替换为其相应的图像以产生合成多模态数据。示例包括将问答数据集中的文本渲染为图像或将表格数据渲染为表格和图表的合成图像。此外，我们还使用现有图像的字幕和 OCR 提取来生成与图像相关的额外对话或问答数据。

视频 类似于图像适配器，我们使用具有预先存在的注释的学术数据集，并将它们转换为适当的文本指令和目标响应。目标被转换为开放式响应或多项选择选项，视情况而定。我们要求人类用问题和相应的答案注释视频。注释者被要求专注于不能仅基于单个帧回答的问题，以引导注释者提出需要时间理解的问题。

7.5.2 监督微调配方

我们分别描述了图像和视频能力的监督微调（SFT）配方。

图像 我们从预训练的图像适配器开始初始化，但将预训练的语言模型权重与指令调整的语言模型权重进行热交换。为了保持仅文本性能，语言模型权重保持冻结，即我们只更新视觉编码器和图像适配器权重。

我们对模型进行微调的方法类似于 Wortsman 等人（2022）。首先，我们使用多个随机数据子集、学习率和权重衰减值进行超参数扫描。接下来，我们根据它们的表现对模型进行排名。最后，我们平均顶级 K 模型的权重以获得最终模型。K 值是通过评估平均模型并选择最高性能的实例来确定的。我们观察到平均模型始终比通过网格搜索找到的最佳单个模型产生更好的结果。此外，这种策略减少了对超参数的敏感性。

视频 对于视频 SFT，我们使用预训练权重初始化视频聚合器和交叉注意力层。模型中的其余参数，图像权重和 LLM，从相应的模型中初始化，跟随它们的微调阶段。类似于视频预训练，我们然后仅在视频 SFT 数据上微调视频参数。在这个阶段，我们将视频长度增加到 64 帧，并使用 32 的聚合因子获得两个有效帧。块的分辨率也增加，以与相应的图像超参数一致。

7.5.3 偏好数据

我们为奖励建模和直接偏好优化构建了多模态成对偏好数据集。

- 人类注释。人类注释的偏好数据包括两个不同模型输出之间的比较，标记为“选择”和“拒绝”，并带有 7 级评分。用于生成响应的模型是从最佳近期模型池中即时采样的，每个模型具有不同的特征。我们每周更新模型池。除了偏好标签，我们还要求注释者提供可选的人类编辑，以纠正“选择”响应中的不准确性，因为视觉任务对不准确性的容忍度很低。注意，人类编辑是可选步骤，因为在实践中存在体积和质量之间的权衡。

- 合成数据。也可以通过使用仅文本 LLM 编辑和故意在监督微调数据集中引入错误来生成合成偏好对。我们将对话数据作为输入，并使用 LLM 引入微妙但有意义的错误（例如，更改对象、更改属性、添加计算错误等）。这些编辑后的响应用作负面的“拒绝”样本，并与“选择”原始监督微调数据配对。

- 拒绝采样。此外，为了创建更多的策略内负样本，我们利用拒绝采样的迭代过程收集额外的偏好数据。我们在以下部分更详细地讨论我们对拒绝采样的使用。在高层次上，拒绝采样用于迭代地从模型中采样高质量生成。因此，作为副产品，所有未被选择的生成都可以用作负面拒绝样本，并用作额外的偏好数据对。

7.5.4 奖励建模

我们在视觉 SFT 模型和语言 RM 的基础上训练一个视觉奖励模型（RM）。视觉编码器和交叉注意力层从视觉 SFT 模型中初始化并在训练期间解冻，而自注意力层从语言 RM 中初始化并保持冻结。我们观察到冻结语言 RM 部分通常可以带来更高的准确性，特别是在需要 RM 根据其知识或语言质量进行判断的任务上。我们采用与语言 RM 相同的训练目标，但增加了一个加权正则化项，该项作用于批次平均奖励 logits 的平方，这防止了奖励分数的漂移。

第 7.5.3 节中的人类偏好注释用于训练视觉 RM。我们按照语言偏好数据（第 4.2.1 节）的相同做法，创建两到三对具有明确排名的配对（编辑 > 选择 > 拒绝）。此外，我们还通过扰乱与图像中的信息（如数字或视觉文本）相关的单词或短语，来增强负面响应。这鼓励视觉 RM 根据实际图像内容做出判断。

7.5.5 直接偏好优化

与语言模型（第 4.1.4 节）类似，我们使用第 7.5.3 节中描述的偏好数据，通过直接偏好优化（DPO；Rafailov 等人，2023）进一步训练视觉适配器。为了对抗后训练轮次中的分布偏移，我们只保留最近的人类偏好注释批次，同时丢弃那些足够偏离策略的批次（例如，如果基础预训练模型发生了变化）。我们发现，不是始终冻结参考模型，而是每 k 步以指数移动平均（EMA）的方式更新它，有助于模型更多地从数据中学习，从而在人类评估中获得更好的表现。总体而言，我们观察到视觉 DPO 模型在每次微调迭代的人类评估中的表现始终优于其 SFT 起点。

7.5.6 拒绝采样

大多数可用的问答对只包含最终答案，缺少训练一个能够很好地泛化推理任务的模型所需的思维链解释。我们使用拒绝采样为这些例子生成缺失的解释，并增强模型的推理能力。

给定一个问题-答案对，我们通过使用不同的系统提示或温度对微调模型进行采样，生成多个答案。接下来，我们通过启发式或 LLM 裁判将生成的答案与真实答案进行比较。最后，我们通过将正确答案重新添加到微调数据混合中来重新训练模型。我们发现保留每个问题的多个正确答案很有用。

为确保我们只将高质量的示例重新添加到训练中，我们实施了以下两个保护措施。首先，我们发现有些示例尽管最终答案正确，但包含的解释却是错误的。我们观察到这种模式更频繁地出现在只有一小部分生成答案正确的问题上。因此，我们放弃了那些答案正确概率低于某个阈值的问题的答案。其次，评估者由于语言或风格的差异而偏好某些答案。我们使用奖励模型选择最高质量答案的前 K 名，并将它们重新添加到训练中。

7.5.7 质量调整

我们策划了一个小而高度精选的 SFT 数据集，所有样本都经过人类或我们最好的模型重写和验证，以满足我们的最高标准。我们使用这些数据训练 DPO 模型以提高响应质量，将这个过程称为质量调整（QT）。我们发现，当 QT 数据集涵盖了广泛的任务并且应用了适当的早期停止时，QT 显著提高了人类评估的表现，而不影响通过基准测试验证的泛化能力。在这个阶段，我们仅基于基准测试选择检查点，以确保能力得以保留或提高。

7.6 图像识别结果

我们在一系列任务上评估了 Llama 3 图像理解能力的表现，这些任务涵盖了自然图像理解、文本理解、图表理解和多模态推理：

- MMMU (Yue 等人，2024a) 是一个具有挑战性的数据集，用于多模态推理，期望模型理解图像并解决涵盖 30 个不同学科的大学水平问题。这包括多项选择和开放式问题。我们按照其他作品的做法，在包含 900 张图像的验证集上评估我们的模型。
- VQAv2 (Antol 等人，2015) 测试模型结合图像理解、语言理解和常识来回答关于自然图像的通用问题的能力。
- AI2 Diagram (Kembhavi 等人，2016) 评估模型解析科学图表并回答相关问题的能力。我们使用与 Gemini 和 x.ai 相同的评估协议，并使用透明边框报告分数。
- ChartQA (Masry 等人，2022) 是一个具有挑战性的图表理解基准。这要求模型直观地理解不同类型的图表并回答有关图表的逻辑问题。
- TextVQA (Singh 等人，2019) 是一个流行的基准数据集，要求模型读取并推理图像中的文本以回答有关它们的问题。这测试了模型在自然图像上的 OCR 理解能力。
- DocVQA (Mathew 等人，2020) 是一个专注于文档分析和识别的基准数据集。它包含各种文档的图像，评估模型执行 OCR 理解和推理文档内容以回答问题的能力。

表 29 展示了我们的实验结果。表中的结果表明，附加到 Llama 3 的视觉模块在不同模型容量的广泛图像识别基准测试中表现出竞争力。使用我们生成的 Llama 3-V 405B 模型，我们在所有基准测试中都优于 GPT-4V，虽然略逊于 Gemini 1.5 Pro 和 Claude 3.5 Sonnet。Llama 3 405B 在文档理解任务上特别具有竞争力。

7.7 视频识别结果

我们在三个基准测试上评估了 Llama 3 的视频适配器：

	Llama 3-V 8B	Llama 3-V 70B	Llama 3-V 405B	GPT-4V	GPT-4o	Gemini 1.5 Pro	Claude 3.5
MMMU (val, CoT)	49.6	60.6	64.5	56.4	**69.1**	62.2	68.3
VQAv2 (test-dev)	78.0	79.1	**80.2**	77.2	–	**80.2**	–
AI2 Diagram (test)	84.4	93.0	94.1	78.2	94.2	94.4	**94.7**
ChartQA (test, CoT)	78.7	83.2	85.8	78.4	85.7	87.2	**90.8**
TextVQA (val)	78.2	83.4	**84.8**	78.0	–	78.7	–
DocVQA (test)	84.4	92.2	92.6	88.4	92.8	93.1△	**95.2**

图 24: 表 29 我们附加到 Llama 3 的视觉模块的图像理解性能。我们将模型性能与 GPT-4V、GPT-4o、Gemini 1.5 Pro 和 Claude 3.5 Sonnet 进行比较。使用外部 OCR 工具获得的结果。

- PerceptionTest (Pătrăucean 等人，2023) 评估模型回答关注技能（记忆、抽象、物理、语义）和不同类型的推理（描述性、解释性、预测性、反事实）的时间推理问题的能力。它包含 11.6K 个测试 QA 对，每个视频平均 23 秒长，由全球 100 名参与者拍摄，展示感知上有趣的任务。我们专注于多项选择问答任务，每个问题都有三个可能的选项。我们通过将我们的预测提交到在线挑战服务器来报告在保留的测试分割上的表现。

- NExT-QA (Xiao 等人，2021) 是另一个时间和因果推理基准，侧重于开放式问答。它包含 1K 个测试视频，每个平均 44 秒长，搭配 9K 个问题。评估是通过将模型的响应与真实答案使用 Wu-Palmer 相似度（WUPS）(Wu 和 Palmer，1994) 进行比较来执行的。

- TVQA (Lei 等人，2018) 评估模型执行组合推理的能力，需要时空定位相关时刻、识别视觉概念和与基于字幕的对话联合推理。这个数据集源自流行的电视节目，此外还测试了模型利用其对这些电视节目的外部知识回答问题的能力。它包含超过 15K 个验证 QA 对，每个相应的视频片段平均 76 秒长。它还遵循多项选择格式，每个问题有五个选项，我们按照先前的工作（OpenAI，2023b）在验证集上报告性能。

- ActivityNet-QA (Yu 等人，2019) 评估模型对长视频片段进行推理以理解动作、空间关系、时间关系、计数等的能力。它包含来自 800 个视频的 8K 个测试 QA 对，每个视频平均 3 分钟长。对于评估，我们遵循先前工作的协议（Google，2023；Lin 等人，2023；Maaz 等人，2024），其中模型生成短的一词或短语答案，并通过 GPT-3.5 API 评估输出的正确性，将其与真实答案进行比较。我们报告 API 评估的平均准确率。

在进行推理时，我们从完整视频片段中均匀采样帧，并将这些帧与简短的文本提示一起输入模型。由于我们的大多数基准测试涉及回答多项选择问题，我们使用以下提示：从以下选项中选择正确答案：{问题} 用正确的选项字母回答,其他什么都不要写。对于那些需要产生简短答案的基准测试(例如,ActivityNet-QA 和 NExT-QA)，我们使用以下提示：用一个词或短语回答问题。{问题} 对于 NExT-QA，由于评估指标（WUPS）对长度和使用的具体单词敏感，我们另外提示模型要具体，并以最突出的答案回应，例如，在被问及位置问题时，指定 “客厅” 而不是简单地回应 “房子”。对于那些包含字幕的基准测试（即，TVQA），我们在推理期间在提示中包含片段对应的字幕。

我们在表 30 中展示了 Llama 3 8B 和 70B 的性能。我们将 Llama 3 的性能与两个 Gemini 和两个 GPT-4 模型进行了比较。请注意，由于我们在训练或微调数据中没有包含这些基准测试的任何部分，因

此我们所有的结果都是零样本的。我们发现，在后训练期间训练了小型视频适配器的 Llama 3 模型非常有竞争力，甚至在某些情况下比其他可能从预训练开始就利用原生多模态处理的模型更好。鉴于我们只评估了 8B 和 70B 参数模型，Llama 3 在视频识别上表现尤为出色。Llama 3 在 PerceptionTest 上表现最佳，表明该模型具有执行复杂时间推理的强能力。在像 ActivityNet-QA 这样的长篇活动理解任务上，即使模型只处理到 64 帧，Llama 3 也能获得强大的结果，这意味着对于 3 分钟长的视频，模型每 3 秒只处理一帧。

	Llama 3-V 8B	Llama 3-V 70B	Gemini 1.0 Pro	Gemini 1.0 Ultra	Gemini 1.5 Pro	GPT-4V	GPT-4o
PerceptionTest (test)	53.8	**60.8**	51.1	54.7	–	–	–
TVQA (val)	82.5	**87.9**	–	–	–	87.3	–
NExT-QA (test)	27.3	**30.3**	28.0	29.9	–	–	–
ActivityNet-QA (test)	52.7	56.3	49.8	52.2	57.5	–	**61.9**

图 25: 表 30 我们附加到 Llama 3 的视觉模块的视频理解性能。我们发现，在涵盖长篇和时间视频理解的任务范围内，我们为 Llama 3 的 8B 和 70B 参数的视觉适配器具有竞争力，有时甚至超越了其他模型。

8 Speech Experiments

我们进行实验，研究将语音能力以组合方式整合到 Llama 3 中的方法，类似于我们用于视觉识别的方法。在输入端，一个编码器和一个适配器被结合用来处理语音信号。我们利用文本形式的系统提示来启用 Llama 3 中不同的语音理解模式。如果没有提供系统提示，模型作为一个通用的口语对话模型，能够以与 Llama 3 文本版本一致的方式有效响应用户的语音。对话历史被引入作为提示前缀，以改善多轮对话体验。我们还尝试使用系统提示，启用 Llama 3 用于自动语音识别（ASR）和自动语音翻译（AST）。Llama 3 的语音接口支持多达 34 种语言。它还允许文本和语音的交错输入，使模型能够解决高级的音频理解任务。

我们还尝试了一种语音生成方法，我们实现了一个流式文本到语音（TTS）系统，在语言模型解码期间即时生成语音波形。我们为 Llama 3 设计的语音生成器基于专有的 TTS 系统，并没有为语音生成微调语言模型。相反，我们专注于通过在推理时利用 Llama 3 嵌入来提高语音合成的延迟、准确性和自然性。语音界面在图 28 和 29 中说明。

8.1 数据

8.1.1 语音理解

训练数据可以分为两种类型。预训练数据包括大量未标记的语音，用于以自监督方式初始化语音编码器。监督微调数据包括语音识别、语音翻译和口语对话数据；当与大型语言模型集成时，这些数据用于解锁特定能力。

预训练数据 为了预训练语音编码器，我们策划了一个大约 1500 万小时的语音录音数据集，涵盖了许多语言。我们使用语音活动检测（VAD）模型过滤我们的音频数据，并选择 VAD 阈值高于 0.7 的音频样

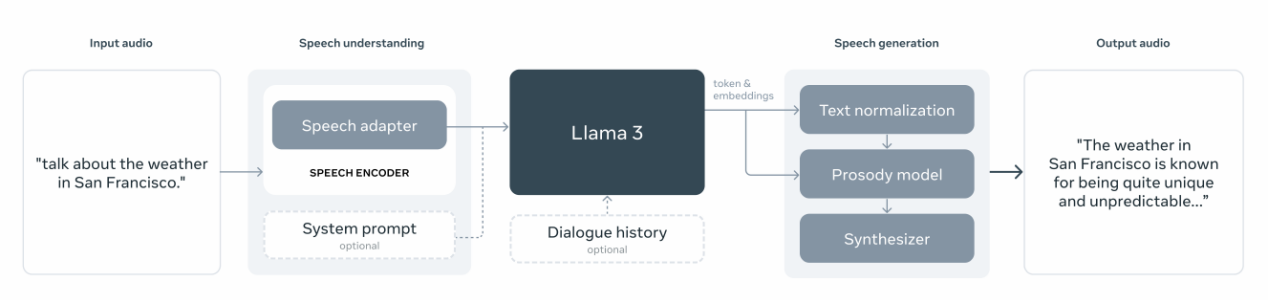

图 26: 图 29 Llama 3 的语音接口架构。

本进行预训练。在语音预训练数据中，我们还专注于确保没有个人身份信息（PII）。我们使用 Presidio Analyzer 来识别此类 PII。

语音识别和翻译数据 我们的 ASR 训练数据包含 23 万小时的手动转录语音录音，涵盖 34 种语言。我们的 AST 训练数据包含 9 万小时的翻译，两个方向：从 33 种语言到英语，以及从英语到 33 种语言。这些数据包含使用 NLLB 工具包（NLLB 团队等人，2022）生成的监督和合成数据。使用合成 AST 数据使我们能够提高低资源语言的模型质量。我们数据中的语音段的最大长度为 60 秒。

口语对话数据 为了微调口语对话的语音适配器，我们通过要求语言模型响应这些提示的转录，合成地生成语音提示的响应（Fathullah 等人，2024）。我们以这种方式使用 ASR 数据集的子集生成 60,000 小时的合成数据。此外，我们通过在用于微调 Llama 3 的数据子集上运行 Voicebox TTS 系统（Le 等人，2024），生成了 25,000 小时的合成数据。我们使用几种启发式方法选择与语音分布匹配的微调数据子集。这些启发式方法包括专注于相对简短的提示，具有简单结构，且没有非文本符号。

8.1.2 语音生成

语音生成数据集主要包括用于训练文本规范化（TN）模型和韵律模型（PM）的数据。两种训练数据都通过 Llama 3 嵌入的额外输入特征增强，以提供上下文信息。

文本规范化数据 我们的 TN 训练数据集包括 55,000 个样本，涵盖了需要非平凡规范化的广泛符号类别（例如，数字、日期、时间）。每个样本是书面形式文本和相应的规范化口语形式文本的配对，以及执行规范化的手工 TN 规则的推断序列。

韵律模型数据 PM 训练数据包括从 50,000 小时 TTS 数据集中提取的语言学和韵律特征，这些是由专业配音演员在录音室环境中录制的配对文稿和音频。

Llama 3 嵌入 Llama 3 嵌入被取作第 16 个解码器层的输出。我们专门使用 Llama 3 8B 模型，并提取给定文本的嵌入（即 TN 的书面形式输入文本或 PM 的音频文稿），就好像它们是由带有空用户提示的 Llama 3 模型生成的。在给定样本中，Llama 3 令牌序列中的每个块都与 TN 或 PM 的原生输入序列中的相应块明确对齐，即，TN 特定的文本令牌（由 unicode 类别分隔）或电话速率特征。这允许使用 Llama 3 令牌和嵌入的流输入来训练 TN 和 PM 模块。

8.2 模型架构

8.2.1 语音理解

在输入端，语音模块由两个连续的模块组成：语音编码器和适配器。语音模块的输出直接输入到语言模型作为标记表示，实现语音和文本标记之间的直接交互。此外，我们引入了两个新的特殊的标记来包围语音表示序列。语音模块与视觉模块（见第 7 节）有很大的不同，后者通过交叉注意力层将多模态信息输入到语言模型中。相比之下，语音模块生成的嵌入可以无缝地与文本标记集成，使语音接口能够利用 Llama 3 语言模型的所有能力。

语音编码器 我们的语音编码器是一个具有 10 亿参数的 Conformer 模型。模型的输入由 80 维的 mel 频谱图特征组成，首先由一个步长为 4 的堆叠层处理，然后通过线性投影将帧长度减少到 40 毫秒。然后由 24 层 Conformer 层处理得到的特征。每个 Conformer 层具有 1536 的潜在维度，并包含两个 Macron-net 风格的前馈网络，其维度为 4096，一个卷积模块，核大小为 7，以及一个旋转注意力模块（Su 等人，2024）和 24 个注意力头。

语音适配器 语音适配器包含大约 1 亿参数。它由一个卷积层、一个旋转 Transformer 层和一个线性层组成。卷积层的核大小为 3，步长为 2，旨在将语音帧长度减少到 80 毫秒。这允许模型向语言模型提供更粗粒度的特征。Transformer 层具有 3072 的潜在维度和一个 4096 维度的前馈网络，这在卷积下采样后进一步处理语音信息和上下文。最后，线性层将输出维度映射以匹配语言模型嵌入层。

8.2.2 语音生成

我们在语音生成的两个关键组件中使用 Llama 3 8B 嵌入：文本规范化和韵律建模。TN 模块通过上下文转换书面文本为口语形式，确保生成语音的语义正确性。PM 模块通过使用这些嵌入预测韵律特征来增强自然性和表现力。它们共同实现准确和自然的语音生成。

文本规范化 作为生成语音语义正确性的决定因素，文本规范化（TN）模块执行从书面形式文本到最终由下游组件口头表达的口语形式的上下文感知转换。例如，书面形式文本 123 根据语义上下文被读作基数词（一百二十三）或逐位拼写（一二三）。TN 系统由一个流式 LSTM 基础序列标记模型组成，该模型预测用于转换输入文本的一系列手工 TN 规则（Kang 等人，2024）。神经模型还通过交叉注意力接收 Llama 3 嵌入，利用其中编码的上下文信息，实现最小的文本标记前瞻和流式输入/输出。

韵律建模 为了增强合成语音的自然性和表现力，我们集成了一个仅解码器的基于 Transformer 的韵律模型（PM）（Radford 等人，2021），它将 Llama 3 嵌入作为额外的输入。这种集成利用了 Llama 3 的语言能力，使用其文本输出和中间嵌入在标记速率（Devlin 等人，2018；Dong 等人，2019；Raffel 等人，2020；Guo 等人，2023）来增强韵律特征的预测，从而减少了模型所需的前瞻。

PM 集成了几个输入组件以生成全面的韵律预测：从上述文本规范化前端派生的语言学特征、标记和嵌入。PM 预测三个关键的韵律特征：每个音素的对数持续时间、对数基频（基频）平均值以及在整个音素持续时间内的对数功率平均值。模型包括一个单向 Transformer 和六个注意力头。每个块包括交叉注意力层和两个具有 864 隐藏维度的全连接层。PM 的一个显著特点是其双重交叉注意力机制，一层专门用于语言输入，另一层专门用于 Llama 嵌入。这种设置有效地管理了不同的输入速率，而无需显式对齐。

8.3 训练方法

8.3.1 语音理解

语音模块的训练分为两个阶段。第一阶段是语音预训练，利用未标记的数据训练一个在不同语言和声学条件下具有强大泛化能力的语音编码器。第二阶段是监督微调，适配器和预训练编码器与语言模型集成在一起，在 LLM 保持冻结的状态下与它一起训练。这使模型能够响应语音输入。这个阶段使用的是有标记的数据，对应于语音理解能力。

多语种 ASR 和 AST 建模通常会导致语言混淆/干扰，从而导致性能下降。缓解这个问题的一种流行方法是在源语言和目标语言两侧都加入语言识别（LID）信息。这可以在预定的方向上提高性能，但确实会带来潜在的泛化能力损失。例如，如果翻译系统期望在源语言和目标语言两侧都有 LID，那么模型在训练中未见的方向上可能不会表现出良好的零样本性能。因此，我们的挑战是设计一个系统，它在一定程度上允许 LID 信息，但保持模型的通用性，以便我们可以在未见的方向上进行语音翻译。为了解决这个问题，我们设计了系统提示，其中只包含要发出的文本（目标侧）的 LID。这些提示中没有语音输入（源侧）的 LID 信息，这也可能允许它与代码切换的语音一起工作。对于 ASR，我们使用以下系统提示：请用 {language} 语重复我的话：，其中 {language} 来自 34 种语言中的一种（英语、法语等）。对于语音翻译，系统提示是：将以下句子翻译成 {language} 这种设计已被证明在促使语言模型以期望的语言响应方面是有效的。我们在训练和推理中使用了相同的系统提示。

语音预训练 我们使用自监督 BEST-RQ 算法（Chiu 等人，2022）预训练语音编码器。我们对输入的 mel 频谱图应用了 32 帧长度的掩码，掩码概率为 2.5%。如果语音话语超过 60 秒，我们执行 6K 帧的随机裁剪，相当于 60 秒的语音。我们通过将 4 个连续帧堆叠起来，将 320 维向量投影到 16 维空间，并在 8192 个向量的码本内进行余弦相似性度量的最近邻搜索，来量化 mel 频谱图特征。为了稳定预训练，我们采用 16 个不同的码本。投影矩阵和码本是随机初始化的，在模型训练期间不更新。为了效率，仅在掩蔽帧上使用 multi-softmax 损失。编码器用 2,048 个话语的全局批量大小训练了 50 万步。

监督微调 预训练的语音编码器和随机初始化的适配器在监督微调阶段进一步与 Llama 3 一起优化。在这个过程中，语言模型保持不变。训练数据是 ASR、AST 和口语对话数据的混合。Llama 3 8B 的语音模型用 512 个话语的全局批量大小训练了 65 万次更新，初始学习率为 10^{-4}。Llama 3 70B 的语音模型用 768 个话语的全局批量大小训练了 60 万次更新，初始学习率为 4×10^{-5}。

8.3.2 语音生成

为了支持实时处理，韵律模型采用前瞻机制，考虑固定数量的未来音素和变化数量的未来标记。这确保了在处理传入文本时具有一致的前瞻，这对于低延迟语音合成应用至关重要。

训练 我们开发了一种利用因果掩蔽的动态对齐策略，以促进语音合成中的流式处理。这种策略结合了固定数量未来音素的前瞻机制和变化数量未来标记的前瞻机制，与文本规范化期间的分块过程相一致。对于每个音素，标记前瞻包括由块大小定义的最大标记数量，从而实现 Llama 嵌入的变化前瞻，但对音素的前瞻是固定的。

Llama 3 嵌入来自 Llama 3 8B 模型，在韵律模型训练期间保持冻结。输入的音素速率特征包括语言学和说话者/风格可控性元素。模型训练采用每批 1,024 个话语，每个话语的最大长度为 500 个音素。我们使用 9×10^{-4} 的学习率，使用 AdamW 优化器，在前 3,000 次更新进行学习率预热后，按照余弦时间表训练 100 万次更新。

推理 在推理过程中，采用相同的前瞻机制和因果掩蔽策略，以确保训练和实时处理之间的一致性。PM 以流式方式处理传入的文本，逐个音素更新音素速率特征，逐块更新标记速率特征。只有当该块的第一个音素当前时，才更新新的块输入，保持与训练期间的对齐和前瞻。

对于韵律目标预测，我们采用延迟模式方法（Kharitonov 等人，2021），这增强了模型捕获和再现长期韵律依赖性的能力。这种方法有助于合成语音的自然性和表现力，确保了低延迟和高质量的输出。

8.4 语音理解结果

我们在三项任务上评估了 Llama 3 的语音接口的语音理解能力：（1）自动语音识别，（2）语音翻译，以及（3）口语问题回答。我们将 Llama 3 的语音接口的性能与三种语音理解的最先进模型进行比较：Whisper（Radford 等人，2023 年）、SeamlessM4T（Barrault 等人，2023 年）和 Gemini。在所有评估中，我们对 Llama 3 的标记预测使用了贪婪搜索。

语音识别 我们在多语种 LibriSpeech（MLS；Pratap 等人，2020 年）、LibriSpeech（Panayotov 等人，2015 年）、VoxPopuli（Wang 等人，2021a）以及多语种 FLEURS 数据集的一个子集（Conneau 等人，2023 年）的英语数据集上评估了 ASR 性能。在评估中，解码结果使用 Whisper 文本规范化工具进行后处理，以确保与其它模型报告的结果进行一致性比较。在所有基准测试中，我们测量了 Llama 3 的语音接口在这些基准的标准测试集上的词错误率，除了中文、日文、韩文和泰语，这些语言报告的是字符错误率。

	Llama 3 8B	Llama 3 70B	Whisper	SeamlessM4T v2	Gemini 1.0 Ultra	Gemini 1.5 Pro
MLS (English)	4.9	4.4	6.2 (v2)	6.5	4.4	**4.2**
LibriSpeech (test-other)	3.4	**3.1**	4.9 (v2)	6.2	–	–
VoxPopuli (English)	6.2	**5.7**	7.0 (v2)	7.0	–	–
FLEURS (34 languages)	9.6	**8.2**	14.4 (v3)	11.7	–	–

图 27: 表 31 Llama 3 的语音接口在语音识别任务上的词错误率。我们报告了 Whisper、SeamlessM4T 和 Gemini 的性能以供参考。

语音翻译 我们还评估了模型在语音翻译任务上的性能，其中模型被要求将非英语语音翻译成英文文本。我们使用 FLEURS 和 Covost 2（Wang 等人，2021b）数据集进行这些评估，测量翻译成英语的 BLEU 分数。表 32 展示了这些实验的结果。我们模型在语音翻译方面的性能突显了多模态基础模型对于像语音翻译这样的任务的优势。

	Llama 3 8B	Llama 3 70B	Whisper v2	SeamlessM4T v2
FLEURS (33 lang. → English)	29.5	**33.7**	21.9	28.6
Covost 2 (15 lang. → English)	34.4	**38.8**	33.8	37.9

图 28: 表 32 Llama 3 的语音接口在语音翻译任务上的 BLEU 分数。我们报告了 Whisper 和 SeamlessM4T 的性能以供参考。

口语问题回答 Llama 3 的语音接口展示了卓越的问题回答能力。模型能够轻松理解代码切换的语音，而无需事先接触此类数据。值得注意的是，尽管模型仅在单轮对话上进行训练，但它能够参与扩展的、连贯的多轮对话会话。图 30 展示了一些突出这些多语言和多轮对话能力的例子。

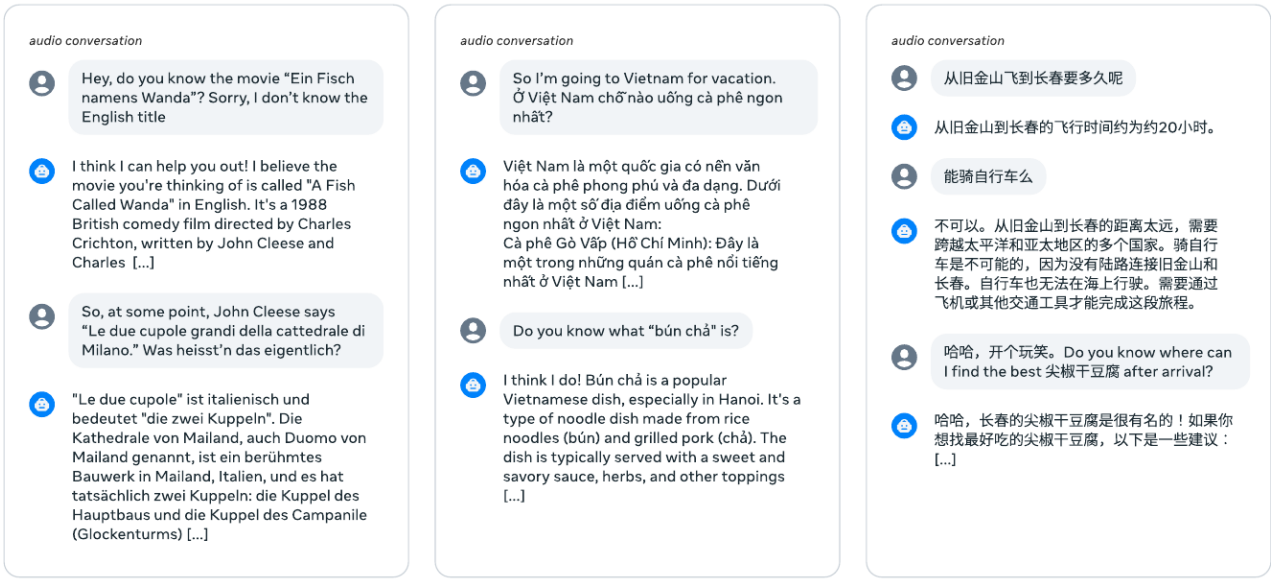

图 29: 图 30 使用 Llama 3 的语音接口转录的对话示例。这些示例展示了零样本多轮对话和代码切换能力。

安全性 我们在 MuTox（Costa-jussà 等人，2023 年）上评估了我们语音模型的安全性，这是一个包含 20,000 条英语和西班牙语发音的多语种音频数据集，另外还有 19 种其他语言的 4,000 条发音，每条都附有毒性标签。音频作为输入传递给模型，输出在清理一些特殊字符后评估其毒性。我们应用了 MuTox 分类器（Costa-jussà 等人，2023 年），并将结果与 Gemini 1.5 Pro 进行了比较。我们评估了添加毒性（AT）的百分比，即输入提示安全而输出有毒时，以及丢失毒性（LT）的百分比，即输入提示有毒而答案安全时。表 33 显示了英语的结果以及我们评估的所有 21 种语言的平均值。添加的毒性百分比非常低：我们的语音模型在英语上添加的毒性百分比最低，不到 1%。它去除的毒性明显多于它增加的毒性。

	Llama 3 8B		Llama 3 70B		Gemini 1.5 Pro	
Language	AT (↓)	LT (↑)	AT (↓)	LT (↑)	AT (↓)	LT (↑)
English	0.84	15.09	**0.68**	**15.46**	1.44	13.42
Overall	2.31	9.89	**2.00**	10.29	2.06	**10.94**

图 30: 表 33 Llama 3 的语音接口在 MuTox 数据集上的语音毒性。AT 指的是添加的毒性百分比，LT 指的是丢失的毒性百分比。

8.5 语音生成结果

对于语音生成，我们专注于评估使用 Llama 3 嵌入进行文本规范化和韵律建模任务的逐令牌输入流式模型的质量。评估的重点是与不将 Llama 3 嵌入作为额外输入的模型进行比较。

文本规范化 为了衡量 Llama 3 嵌入的效果，我们尝试改变模型使用的右侧上下文量。我们训练模型时使用了 3 个 TN 令牌（由 unicode 类别分隔）的右侧上下文。这个模型与不使用 Llama 3 嵌入，使用 3 个令牌的右侧上下文或完整双向上下文的模型进行了比较。正如预期，表 34 显示，对于不使用 Llama 3 嵌入的模型，使用完整右侧上下文可以提高性能。然而，结合了 Llama 3 嵌入的模型超越了所有其他模型，因此能够在不依赖输入中的长上下文的情况下，实现令牌速率的输入/输出流式处理。

Model	Context	Accuracy
Without Llama 3 8B	3	73.6%
Without Llama 3 8B	∞	88.0%
With Llama 3 8B	3	**90.7%**

图 31: 表 34 逐样本的文本规范化（TN）准确率。我们比较有无 Llama 3 8B 嵌入的模型，并使用不同的右侧上下文值。

韵律建模 为了评估我们的韵律模型（PM）与 Llama 3 8B 的性能，我们进行了两组人类评估，比较有无 Llama 3 嵌入的模型。评估者听取不同模型的样本并表明他们的偏好。为了生成最终的语音波形，我们使用了一个内部的基于 transformer 的声学模型（Wu 等人，2021 年），它预测频谱特征，并使用 WaveRNN 神经声码器（Kalchbrenner 等人，2018 年）来生成最终的语音波形。

首先，我们直接与没有 Llama 3 嵌入的流式基线模型进行了比较。在第二项测试中，Llama 3 8B PM 与没有 Llama 3 嵌入的非流式基线模型进行了比较。正如表 35 所示，与流式基线相比，Llama 3 8B PM 有 60% 的时间被偏好，与非流式基线相比，有 63.6% 的时间被偏好，表明在感知质量上有显著改进。Llama 3 8B PM 的关键优势是其逐令牌流式能力（第 8.2.2 节），这在推理期间保持了低延迟。这减少了模型的前瞻要求，与非流式基线相比，实现了更具响应性和实时性的语音合成。总体而言，Llama 3 8B 韵律模型一贯优于基线模型，证明了其在增强合成语音的自然性和表现力方面的有效性。

Model	Preference
PM for Llama 3 8B	**60.0%**
Streaming phone-only baseline	40.0%

Model	Preference
PM for Llama 3 8B	**63.6%**
Non-streaming phone-only baseline	36.4%

图 32: 表 35 韵律建模（PM）评估。左侧：评估者对 Llama 3 8B PM 与仅音素流式基线模型的偏好。右侧：评估者对 Llama 3 8B PM 与非流式仅音素基线模型的偏好。

9 Related Work

Llama 3 的开发建立在对语言、图像、视频和语音的基础模型进行大量先前研究的基础上。全面概述这些工作超出了本文的范围；我们建议读者参考 Bordes 等人（2024 年）；Madan 等人（2024 年）；Zhao 等人（2023a）以获取此类概述。以下，我们简要概述了对 Llama 3 开发产生直接影响的重要工作。

9.1 Language 语言

规模 Llama 3 遵循了在基础模型中不断增加规模应用简单方法的持久趋势。改进源于计算量的增加和数据质量的提升，405B 模型使用的预训练计算预算几乎是 Llama 270B 的五十倍。尽管包含 405B 参数，我们的最大 Llama 3 实际上包含的参数比早期且性能低得多的模型如 PALM（Chowdhery 等人，2023 年）要少，这是由于对缩放定律的更好理解（Kaplan 等人，2020 年；Hoffmann 等人，2022 年）。关于其他前沿模型如 Claude 3 或 GPT 4（OpenAI，2023a）的规模公开信息很少，但总体性能相当。

小型模型。小型模型的发展与大型模型并行。参数较少的模型可以显著降低推理成本并简化部署（Mehta et al., 2024; Team et al., 2024）。较小的 Llama 3 模型通过在计算最优训练点之外进行远超训练，有效地用训练计算换取推理效率。另一种方法是将从大型模型中提炼成小型模型，如 Phi（Abdin et al., 2024）。

架构 尽管 Llama 3 相对于 Llama 2 进行了最小的架构修改，但其他最近的基石模型探索了其他设计。最值得注意的是，专家混合架构（Shazeer et al., 2017; Lewis et al., 2021; Fedus et al., 2022; Zhou et al., 2022）可以作为一种有效的方式来增加模型的容量，例如在 Mixtral（Jiang et al., 2024）和 Arctic（Snowflake, 2024）中。Llama 3 在这些模型中表现更优，表明密集架构并非限制因素，但在训练和推理效率以及大规模模型稳定性方面仍存在许多权衡。

开源 开放权重基础模型在过去一年中迅速改进，现在 Llama3-405B 已经与当前的封闭权重最先进技术相竞争。近期开发了众多模型系列，包括 Mistral（Jiang et al., 2023）、Falcon（Almazrouei et al., 2023）、MPT（Databricks, 2024）、Pythia（Biderman et al., 2023）、Arctic（Snowflake, 2024）、OpenELM（Mehta et al., 2024）、OLMo（Groeneveld et al., 2024）、StableLM（Bellagente et al., 2024）、OpenLLaMA（Geng and Liu, 2023）、Qwen（Bai et al., 2023）、Gemma（Team et al., 2024）、Grok（XAI, 2024）和 Phi（Abdin et al., 2024）。

训练后 Llama 3 遵循既定的策略，即先进行指令调优（Chung et al., 2022; Ouyang et al., 2022），然后通过人类反馈进行对齐（Kaufmann et al., 2023）。尽管一些研究表明轻量级对齐程序具有惊人的有效性（Zhou et al., 2024），但 Llama 3 使用数百万条人类指令和偏好判断来改进预训练模型，包括拒绝采样（Bai et al., 2022）、监督微调（Sanh et al., 2022）和直接偏好优化（Rafailov et al., 2023）等技术。为了策划这些指令和偏好示例，我们部署了早期版本的 Llama 3 进行过滤（Liu et al., 2024c）、重写（Pan et al., 2024）或生成提示和响应（Liu et al., 2024b），并通过多轮训练后应用这些技术。

9.2 Multimodality 多模态

我们对 Llama 3 的多模态能力的实验是关于基础模型联合建模多种模态的长期工作的一部分。

图像 大量工作已经在大规模图像-文本对上训练了图像识别模型，例如，Mahajan 等人（2018）；Xiao 等人（2024a）；Team（2024）；OpenAI（2023b）。Radford 等人（2021）提出了首批通过对比学习联合嵌

入图像和文本的模型之一。最近，一系列模型研究了类似于 Llama 3 中使用的方法，例如，Alayrac 等人（2022）；Dai 等人（2023）；Liu 等人（2023c,b）；Yang 等人（2023b）；Ye 等人（2023）；Zhu 等人（2023）。我们在 Llama 3 中的方法结合了这些论文中的许多思想，实现了与 Gemini 1.0 Ultra（Google，2023）和 GPT-4 Vision（OpenAI，2023b）相当的结果；参见第 7.6 节。

视频 尽管越来越多的基础模型支持视频输入（Google，2023；OpenAI，2023b），但关于视频和语言联合建模的工作量并不大。与 Llama 3 类似，当前大多数研究采用适配器方法来对齐视频和语言表示，并解锁关于视频的问答和推理（Lin 等人，2023；Li 等人，2023a；Maaz 等人，2024；Zhang 等人，2023；Zhao 等人，2022）。我们发现这些方法产生的结果与最先进水平相当；参见第 7.7 节。

语音 我们的工作也符合将语言和语音建模结合起来的更大范围的工作。早期的文本和语音联合模型包括 AudioPaLM（Rubenstein 等人，2023）、VioLA（Wang 等人，2023b）、VoxtLM Maiti 等人（2023）、SUTLM（Chou 等人，2023）和 Spirit-LM（Nguyen 等人，2024）。我们的工作建立在先前结合语音和语言的组合方法上，如 Fathullah 等人（2024）。与大多数先前的工作不同，我们选择不对语言模型本身进行语音任务的微调，因为这样做可能会导致非语音任务上的冲突。我们发现，在更大的模型规模下，即使没有这种微调，也能达到强大的性能；参见第 8.4 节。

10 Conclusion 结论

在许多方面，高质量基础模型的发展仍处于起步阶段。我们在开发 Llama 3 的经验表明，这些模型的进一步重大改进即将到来。在整个 Llama 3 模型家族的开发过程中，我们发现对高质量数据、规模和简单性的强烈关注始终能产生最佳结果。在初步实验中，我们探索了更复杂的模型架构和训练方案，但没有发现这些方法的好处能超过它们在模型开发中引入的额外复杂性。

开发像 Llama 3 这样的旗舰基础模型不仅涉及克服众多深层次技术问题，还需要巧妙的组织决策。例如，为了确保 Llama 3 不会意外地过度适应常用基准测试，我们的预训练数据由一个单独的团队采购和处理，该团队受到强烈激励以防止预训练数据受到外部基准的污染。再比如，我们通过只允许一小部分不参与模型开发的研究人员进行并访问这些评估，来确保我们的人工评估保持可信度。尽管这类组织决策在技术论文中很少被讨论，但我们发现它们对 Llama 3 系列模型的成功开发至关重要。

我们分享了开发过程的细节，因为我们相信这将：（1）帮助更大的研究社区理解基础模型开发的关键因素；（2）有助于公众对基础模型未来进行更明智的辩论。我们还分享了将多模态能力集成到 Llama 3 中的初步实验。虽然这些模型仍在积极开发中，尚未准备好发布，但我们希望早期分享我们的结果将加速这一方向的研究。

基于本文详细安全分析的积极结果，我们公开发布了 Llama 3 语言模型，以加速开发对社会相关用例有广泛影响的 AI 系统，并使研究社区能够仔细审查我们的模型并找出使这些模型更好、更安全的方法。我们相信，基础模型的公开发布在负责任的模型开发中起着关键作用，我们希望 Llama 3 的发布能鼓励业界接受开放、负责任的 AGI 开发。