

Llama 3 Herd of Models

9/10 Advances in Self-Supervised Models

Llama 3.1-405b Instruct is Currently the Best Open Source AI

Rank* (UB) ▲	Model ▲	Arena Score ▲	95% CI ▲	Votes ▲	Organization ▲	License ▲	Knowledge Cutoff ▲
1	ChatGPT-4o-latest (2024-08-08)	1316	+4/-3	31148	OpenAI	Proprietary	2023/10
2	Gemini-1.5-Pro-Exp-0827	1300	+4/-4	22844	Google	Proprietary	2023/11
2	Gemini-1.5-Pro-Exp-0801	1298	+4/-4	26110	Google	Proprietary	2023/11
2	Grok-2-08-13	1294	+4/-4	16215	xAI	Proprietary	2024/3
5	GPT-4o-2024-05-13	1285	+3/-2	86306	OpenAI	Proprietary	2023/10
6	GPT-4o-mini-2024-07-18	1274	+4/-4	26088	OpenAI	Proprietary	2023/10
6	Claude 3.5 Sonnet	1270	+3/-3	56674	Anthropic	Proprietary	2024/4
6	Gemini-1.5-Flash-Exp-0827	1268	+5/-4	16780	Google	Proprietary	2023/11
6	Grok-2-Mini-08-13	1267	+4/-4	16731	xAI	Proprietary	2024/3
6	Meta-Llama-3.1-405b-Instruct	1266	+4/-4	27397	Meta	Llama 3.1 Community	2023/12
7	Gemini Advanced Exp (2024-05-14)	1266	+3/-3	52236	Google	Proprietary	Online
7	GPT-4o-2024-08-06	1263	+4/-4	18093	OpenAI	Proprietary	2023/10

Overview

<https://llama.meta.com/>

- On engineering Llama Models
- Integrates image, video, speech capabilities, natively supports multilinguality, coding, reasoning, and tool usage.
- Not MoEs or some amalgamation of Llamas! **Use a standard dense Transformer.**
- Training Data: 15T / Scale: 8B - 405B
- This presentation is on pre-training only
 - The paper is 92 pages long!



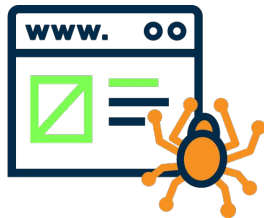
Pre-Training

1. Curation and Filtering of a large-scale training corpus
2. The development of a model architecture and corresponding scaling laws for determining model size
3. The development of techniques for efficient pre-training at a large scale
4. Development of a pre-training recipe



Data Acquisition

Part 1, Pre-training Data



Source

- Much of data obtained from web
- Maybe some Meta APP databases

facebook



Goal

To obtain a high-quality language model

- Obtained multilingual data on general knowledge, code, and reasoning
- Improved the data quality
- Removed some inappropriate text
- Determined the proportion of different data sources

Filtering, Cleaning and Deduplication

Removal of safety filtering

Removed sites that is likely to contain unsafe content, and personal identification info

Text extraction and cleaning

They created HTML parser to remove boilerplate and retrieve content only.

Got rid of markdown markers since it was harmful

expensive!!

Deduplication

Removal of duplicate, old versioned url

Global minhash document-level deduplication

Line level Remove lines that appeared more than 6 times in each bucket of 30M documents



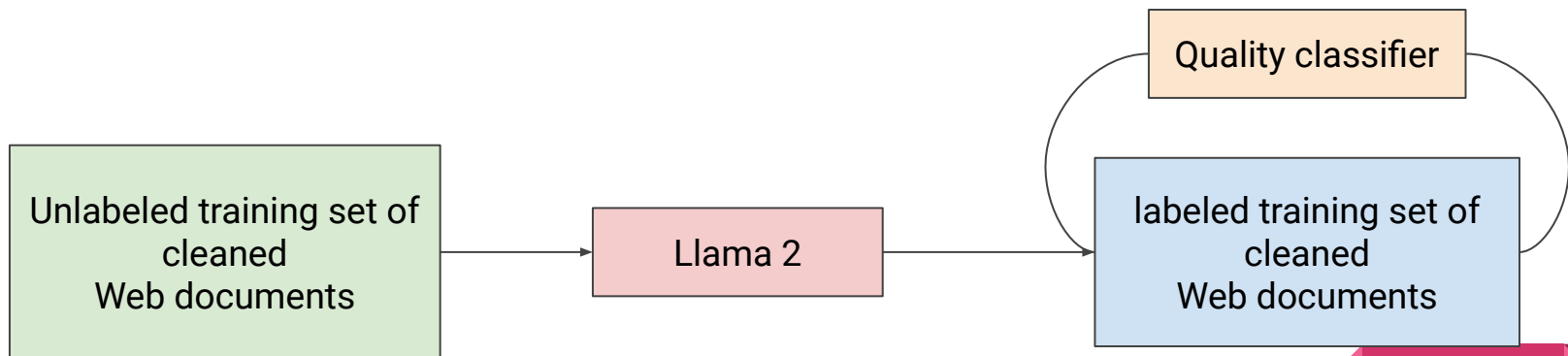
Heuristic Filtering

- Deduplication using n-gram coverage
 - Deduplicates items too long according to line-level deduplication
 - “Dirty Words” counting to remove adult websites
 - Kullback-Leibler divergence to filter documents with excessive outlier tokens



Model-based Quality Filtering

- Fasttext to recognize text referenced by Wikipedia
- Trained a quality classifier DistilRoberta to generated quality scores for each documents




Code and Reasoning Data & Multilingua Data

Code

- Trained a DistilRoberta model based on data annotation of Llama2 for extraction of web pages related to code and math.
- Shares different token distribution from other sites.
 - Required **domain specific** HTML extraction, text features, and heuristics for filtering

Multi-language

- Used **fasttext** based language identification to categorize 176 languages
 - Build custom document-level and line-level deduplication within data for each language
 - Used language-specific filters.
- 

Limitations

- No mention of the actual Data they used. (maybe issues on intellectual property)
- Doesn't specify webpage but the data extraction methods point towards majority web scraping.

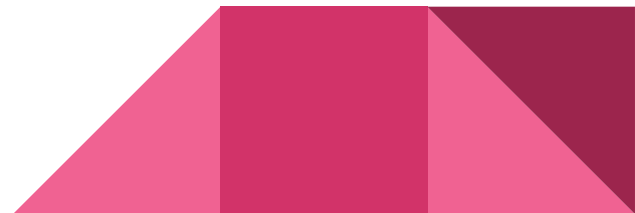


Concoction

- Developed a classifier to categorize the types of information and downsample the categories which is not important like **entertainment**
- Performed **scaling law** experiments in small models to predict patterns on large models.

summary

- 50% - general knowledge
- 42% - **code and reasoning !!!**
- 8% - multilingual



Annealing

- Upsampled high-quality data with a very low learning rate near the end of training, for example in reasoning and math area
- Using in judging the value of specific domain datasets
 - assign 30% weight to new dataset and the remaining 70% weight to the default data mix
 - It can be evaluated whether the datasets has impacts on pre-training





Model

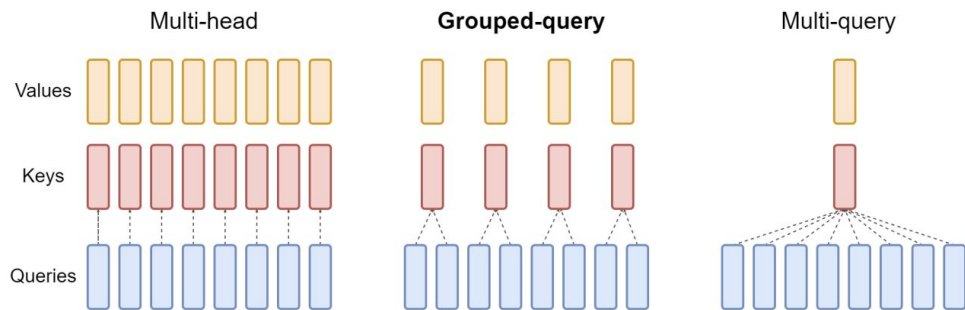
Model Architecture(Compared to Llama 2)

- Grouped query attention(GQA) with 8 key-value heads
- Used attention mask to prevents self-attention between different documents



Grouped query attention(GQA)

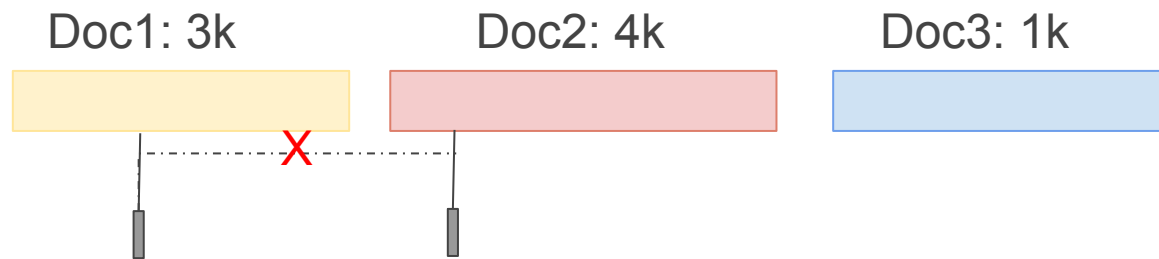
- Divide query heads into $G=8$ groups(each group has $n=4/8/12$ queries), each of which shares a single key head and value head.
- Construct each group key and value head by mean-pooling all the original heads within that group.



- Higher model quality than MPA
- Faster than MHA
- Reduces size of kv-cache

Attention mask

- Significant for continued pre-training on very long sequences



Model Architecture

- Used a vocabulary with 128K tokens. Combined 100K tokens from the tiktoken3 tokenizer with **28K additional tokens to better support non-English languages**
 - tokenizer improves compression rates on a sample of English data from 3.17 to 3.94 characters per token
- Increased the RoPE(position encoding method) base frequency hyperparameter to 500,000 to better support longer contexts.



Model Architecture

	8B	70B	405B
Layers	32	80	126
Model Dimension	4,096	8192	16,384
FFN Dimension	14,336	28,672	53,248
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 ($\theta = 500,000$)	

Scaling Laws

Prev scaling laws (Hoffmann et al., 2022; Kaplan et al., 2020):

- predict only next-token prediction loss rather than **benchmark**
- Scaling laws can be noisy and unreliable because they are developed based on pre-training runs conducted with **small compute budgets**



Scaling Laws

- Established a correlation between the compute-optimal model's negative log-likelihood on **down-stream tasks** and the training FLOPs.
- Correlated the negative log-likelihood on downstream tasks with **task accuracy**, utilizing both the scaling law models and older models trained with higher compute FLOPs. (eg, Llama 2)



Scaling Laws

Optimization

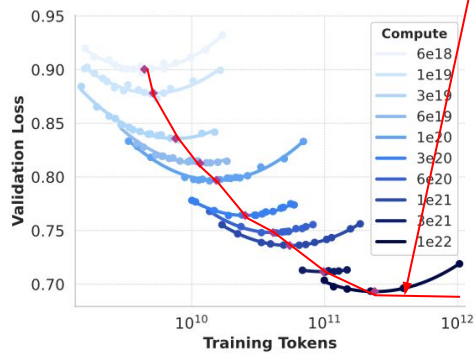


Figure 2 Scaling law IsoFLOPs curves between 6×10^{18} and 10^{22} FLOPs. The loss is the negative log-likelihood on a held-out validation set. We approximate measurements at each compute scale using a second degree polynomial.

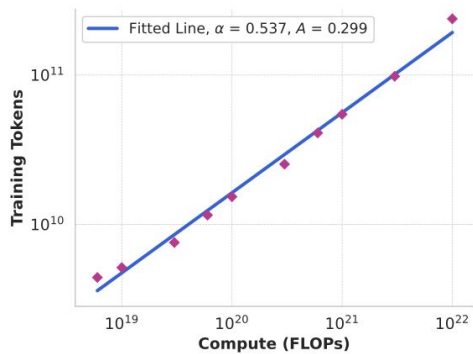
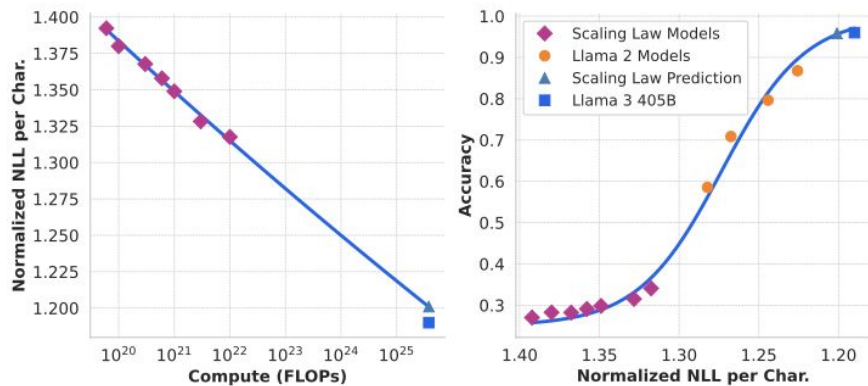


Figure 3 Number of training tokens in identified compute-optimal models as a function of pre-training compute budget. We include the fitted scaling-law prediction as well. The compute-optimal models correspond to the parabola minimums in Figure 2.

Scaling Laws



- Experiments on various benchmark make sense
- Training on small scaling law models and Llama 2 models

Figure 4 Scaling law forecast for ARC Challenge. *Left:* Normalized negative log-likelihood of the correct answer on the ARC Challenge benchmark as a function of pre-training FLOPs. *Right:* ARC Challenge benchmark accuracy as a function of the normalized negative log-likelihood of the correct answer. This analysis enables us to predict model performance on the ARC Challenge benchmark before pre-training commences. See text for details.



Infrastructure, Scaling and Efficiency

Engineering Challenges of Training a 405B Parameter Model

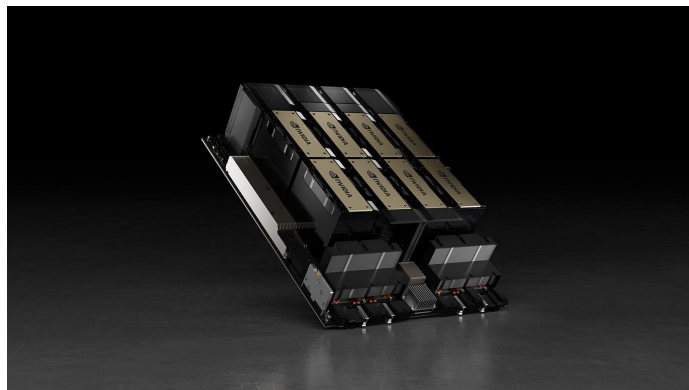
- With 405 billion parameters storing the model weights in full precision (32-bit) results in **1,620 GB** of vram just for the weights
- Optimizers generally use three times the model size of memory. That results in **4,860 GB** of vram
- Training a model of this scale requires **80+ H100 GPUs** to for a single instance
- The model is trained in batches, meaning weight updates must be synchronized across batches, layers, tensors across GPUs.
- Training such model requires massive infrastructure and smart parallelization schemes



Infrastructure

Compute

- 16k H100 80GB GPUs
 - **400 million dollars** on GPU alone
- Server rack consists of eight GPUs and two CPUs
- Training Jobs were scheduled via MAST, Meta's global scale training scheduler

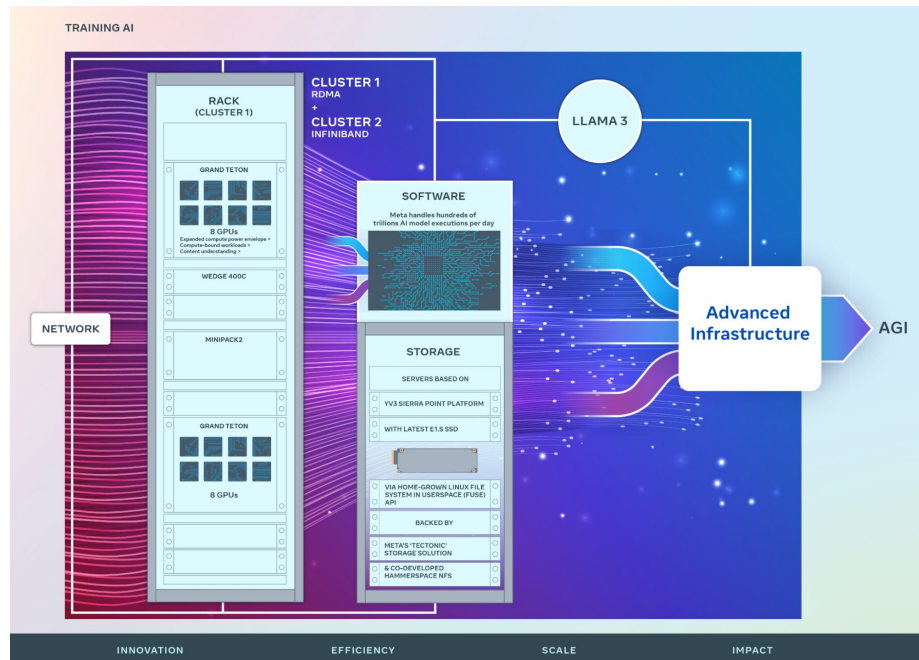


Storage and Network

- Tectonic (Meta's general purpose distributed file system)
 - 240PB of storage, 7,500 servers
 - Each servers have throughput of 2TB/s and peak of 7TB/s
 - High burst occurs on checkpoint save
 - Important to keep frequent checkpoints due to recovery

Network

- Llama 405B used RDMA over Converged Ethernet fabric (RoCE)
- Smaller models in the Llama 3 family were trained using Nvidia Quantum2 Infiniband fabric
- Both clusters leverage 400Gbps interconnects between GPUs

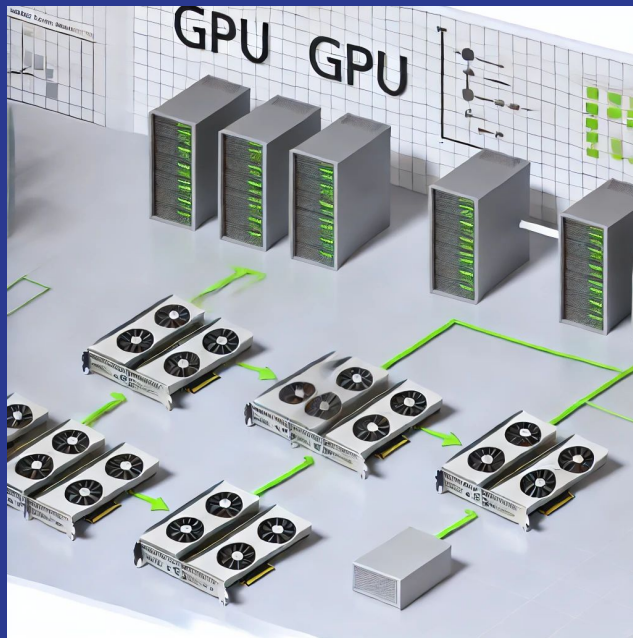


RoCE base structure

- 8 GPUs per rack
- 2 racks per rack host (**16 GPUs**)
- 192 rack hosts per pod(**3072 GPUs**)
- 8 pods (**24K GPUs**)
- Oversubscription ratio of 1:7
- Through load balancing, and congestion control, Llama team tackle the oversubscription ratio
 - Multiple channels between GPUs for granular flow and load balancing
 - Deep Buffer switches in the spine
 - Better load balancing (Check their [paper!](#))



Parallelism



4D Parallelism

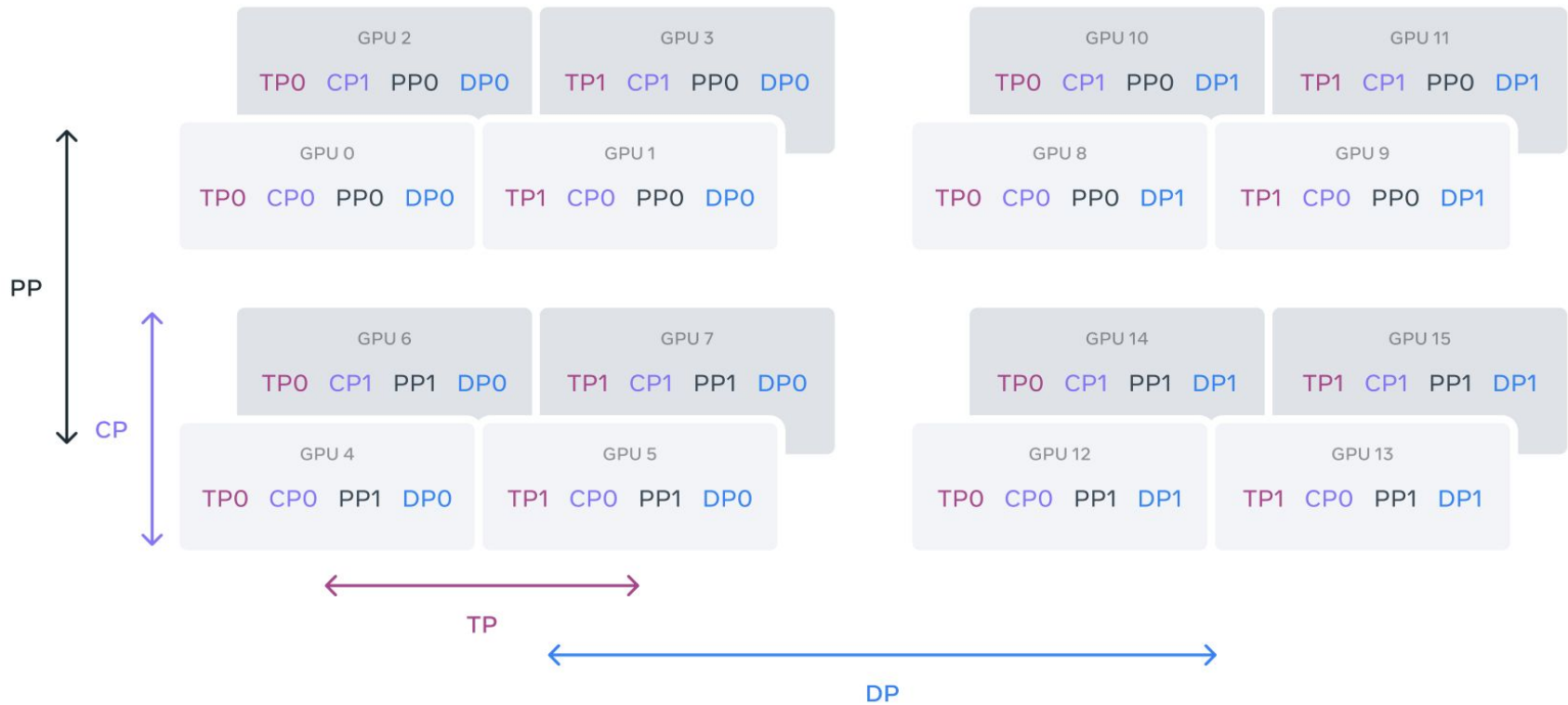
- Tensor Parallelism
 - Chunks weight tensors into different devices
- Context Parallelism
 - Divide input content into segments
- Pipeline Parallelism
 - Partitions model vertically into stages by layers
- Data Parallelism
 - Data parallelism shares the model, optimizer, and gradients



4D Parallelism

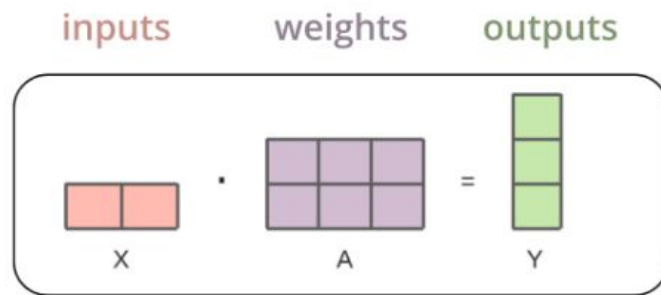
- Tensor Parallelism
- Context Parallelism
- Pipeline Parallelism
- Data Parallelism



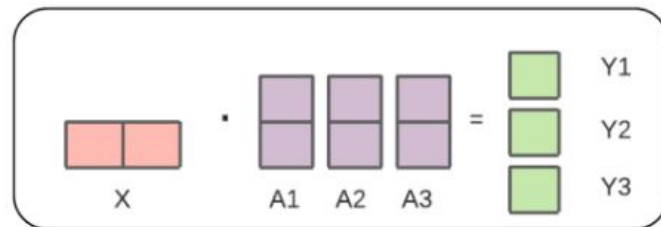


Tensor Parallelism

- Splits large tensors to smaller tensors for modular computation



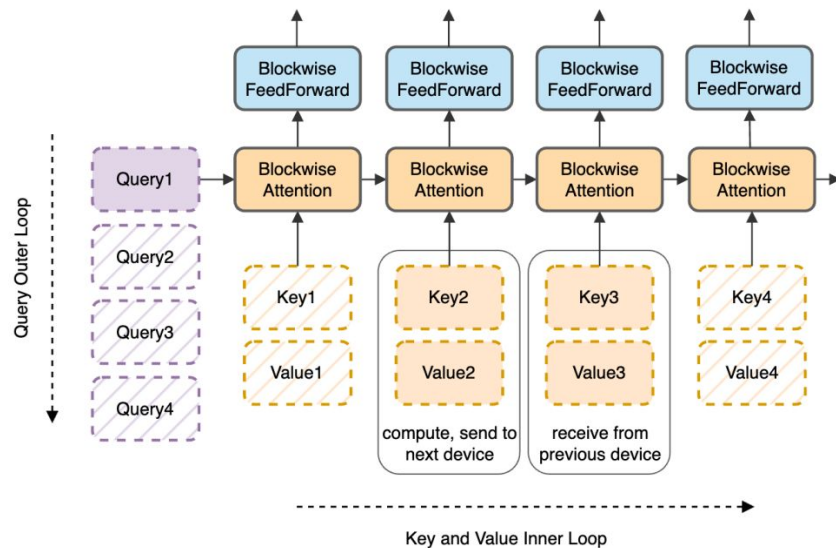
is equivalent to



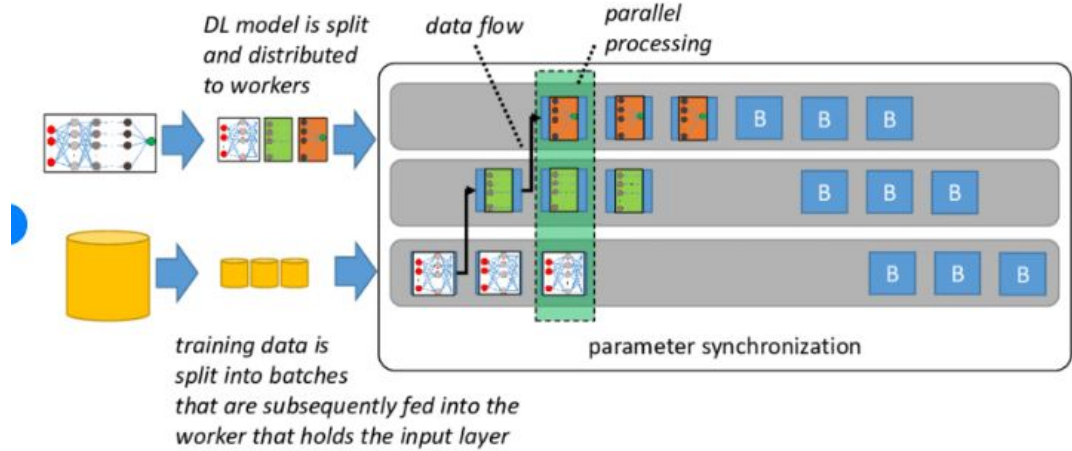
Context Parallelism

- Splits context across multiple GPUs.
- Utilizes ring-attention for compute
 - Utilized compute order flexibility of attention

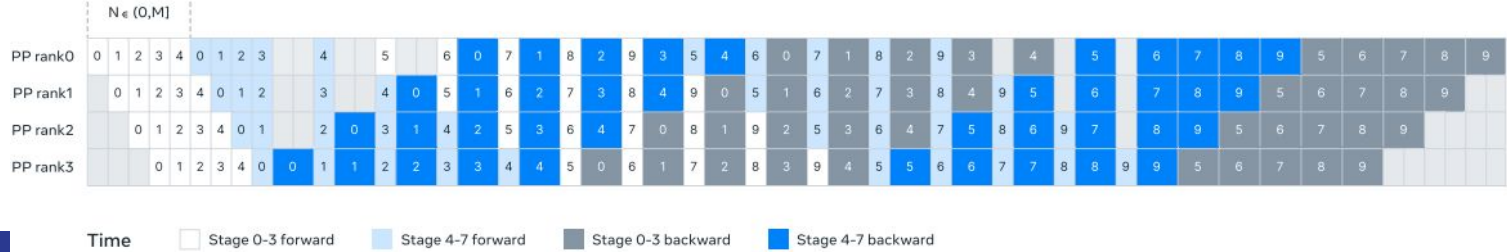
(b)



Pipeline Parallelism



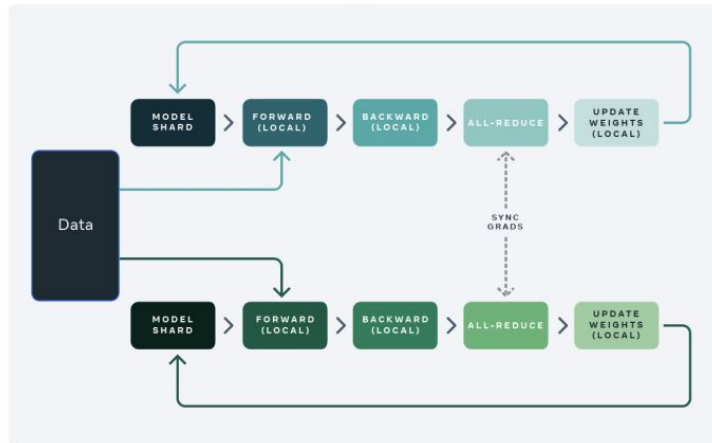
Pipeline parallelism. "B" -Backpropagation. Figure adapted from Huang et al. [70].



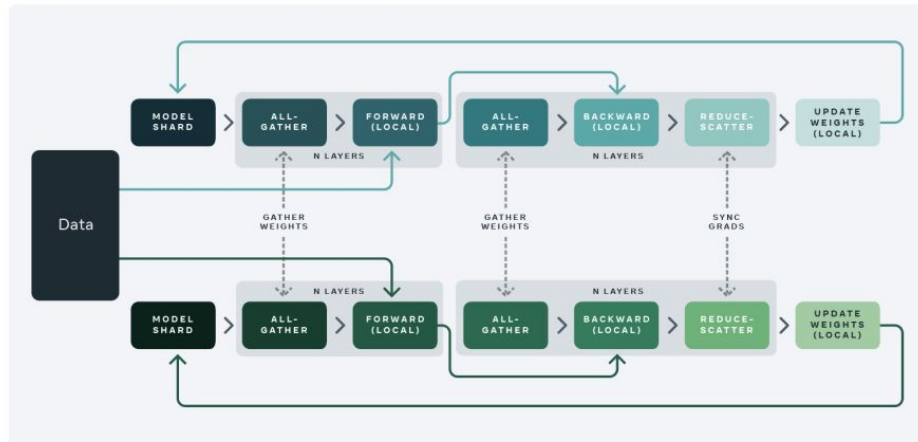
Data Parallelism

- Trains models across different data
- Uses Fully Sharded Data Parallelism (FSDP)
 - Synchronize updates using model, optimizer, and gradients shard.
 - Synchronized update happen every training step

Standard data parallel training



Fully sharded data parallel training



Training Stability

- Achieved **90%** effective training time
- **78%** were due to hardware issues
- Used PyTorch's built in NCCL flight recorder to monitor
- There was a **1-2%** throughput variation based on time of day
 - Result of higher midday temperature

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%

Table 5 Root-cause categorization of unexpected interruptions during a 54-day period of Llama 3 405B pre-training. About 78% of unexpected interruptions were attributed to confirmed or suspected hardware issues.

Training Recipe

Training Recipe

- Initial pre-training
- Long-context pre-training
- Annealing



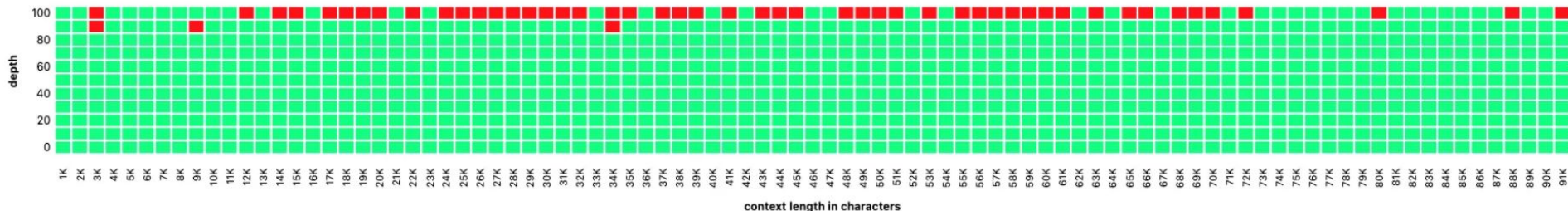
Initial Pre-training

- Used linear warm up of 8,000 steps with peak learning rate of $8e-5$.
- Used cosine learning rate schedule decay to $8e-7$ over 1,200,000 steps
- Used lower batch size early in training to improve training stability
 - 4M sequences of 4096 tokens up until 252M tokens
 - 8M sequences of 8192 tokens until 2.87T tokens
 - Finally train on 16M batch size
- Increased percentage of non-English data and mathematical data for initial pre-training.



Long Context Pre-Training

- Trained on long sequences to support context windows of 128K tokens
- Measured model performance on short-context evals and needle-in-a-haystack task
- Gradually increased 8K context to 128K context window
- 800B training tokens used in long context pre-training



Annealing

- During the final 40M tokens of training, the learning rate is linearly decreased to zero.
- The final pretrained model weights are obtained by averaging over model checkpoints during this period.
- During this period, data mix was enhanced by including text from upsampled data sources of high quality





Pretrained Model Evaluation

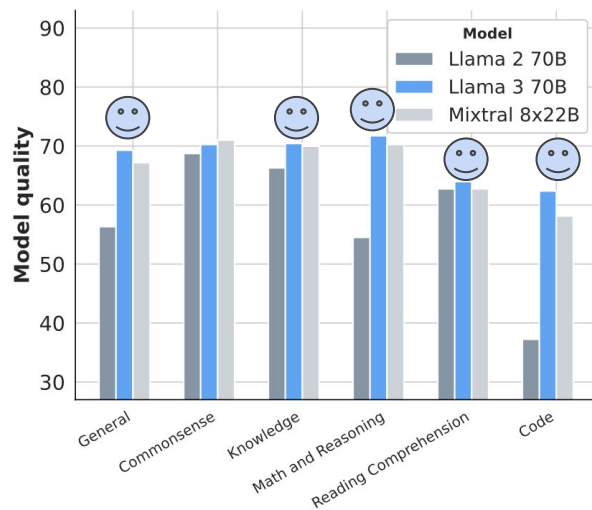
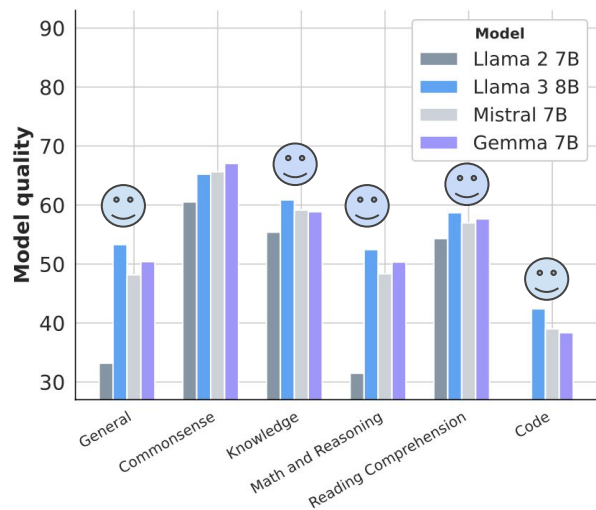
Pretrained Model Results

- Standard Benchmarks
- Robustness to changes in multiple-choice question setups
- adversarial evaluations
- the extent to which our evaluations are impacted by contamination of training data



Standard Benchmarks

Win nearly all types of tasks!!



Standard Benchmarks

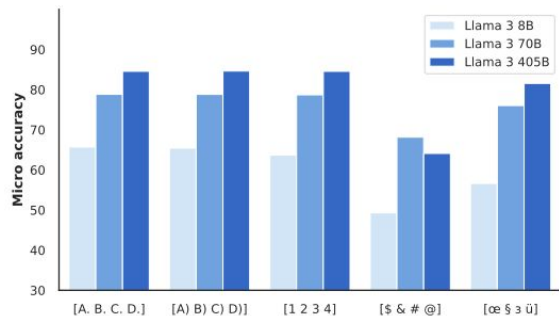
	Math and Reasoning				
	GSM8K	MATH	ARC-C	DROP	WorldSense
Llama 3 8B	57.2 ± 2.7	20.3 ± 1.1	79.7 ± 2.3	59.5 ± 1.0	45.5 ± 0.3
Mistral 7B	52.5 ± 2.7	13.1 ± 0.9	78.2 ± 2.4	53.0 ± 1.0	44.9 ± 0.3
Gemma 7B	46.4 ± 2.7	24.3 ± 1.2	78.6 ± 2.4	56.3 ± 1.0	46.0 ± 0.3
Llama 3 70B	83.7 ± 2.0	41.4 ± 1.4	92.9 ± 1.5	79.6 ± 0.8	61.1 ± 0.3
Mixtral 8 \times 22B	88.4 ± 1.7	41.8 ± 1.4	91.9 ± 1.6	77.5 ± 0.8	51.5 ± 0.3
Llama 3 405B	89.0 ± 1.7	53.8 ± 1.4	96.1 ± 1.1	84.8 ± 0.7	63.7 ± 0.3
GPT-4	92.0 ± 1.5	–	96.3 ± 1.1	80.9 ± 0.8	–
Nemotron 4 340B	–	–	94.3 ± 1.3	–	–
Gemini Ultra	88.9 \diamond ± 1.7	53.2 ± 1.4	–	82.4 Δ ± 0.8	–

405B is very competitive among LM!!

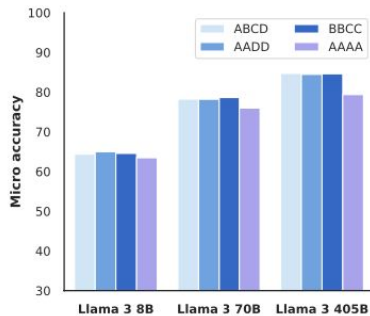
	General			
	MMLU	MMLU-Pro	AGIEval	BB Hard
Llama 3 8B	66.7	37.1	47.8 ± 1.9	64.2 ± 1.2
Mistral 7B	63.6	32.5	42.7 ± 1.9	56.8 ± 1.2
Gemma 7B	64.3	35.1	46.0 ± 1.9	57.7 ± 1.2
Llama 3 70B	79.3	53.8	64.6 ± 1.9	81.6 ± 0.9
Mixtral 8 \times 22B	77.8	51.5	61.5 ± 1.9	79.5 ± 1.0
Llama 3 405B	85.2	61.6	71.6 ± 1.8	85.9 ± 0.8
GPT-4	86.4	–	–	–
Nemotron 4 340B	81.1	–	–	85.4 ± 0.9
Gemini Ultra	83.7	–	–	83.6 ± 0.9

Robustness to changes in MCQ setups

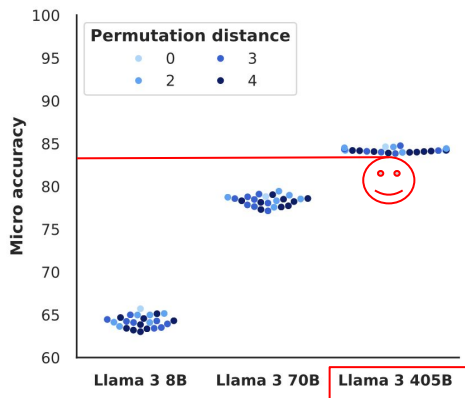
Label variants



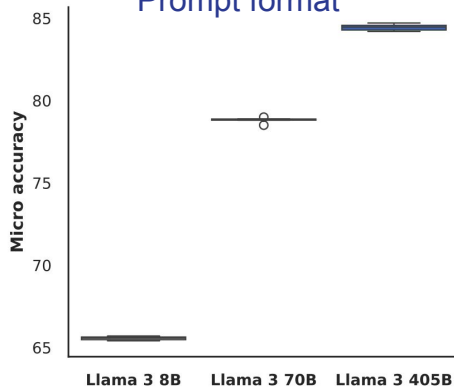
Few-shot label bias



Answer order



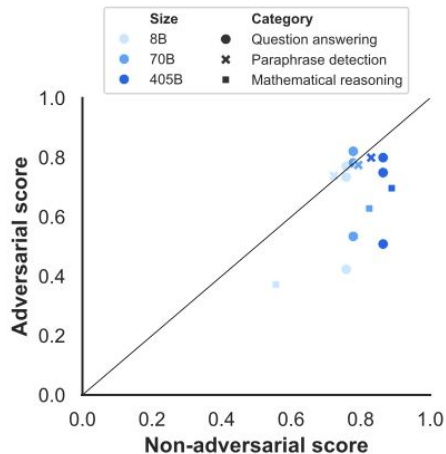
Prompt format



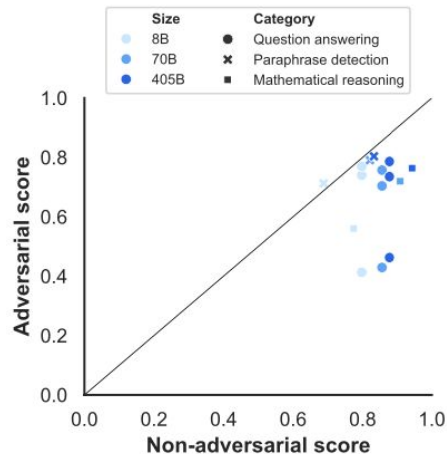
405B model shows high robustness in answer order changes

Adversarial evaluations

similar performance



Results for pre-trained models



Results for post-trained models

the adversarial performances are substantially lower than the non-adversarial performances for math and question answering

Contamination of training data impact

Method:

- Consider an example of a dataset D to be contaminated if a ratio \mathcal{T}_D of its tokens are part of an **8-gram** occurring at least once in the pre-training corpus.
 - Select \mathcal{T}_D separately for each dataset, based on which value shows the maximal significant estimated performance gain across the three model sizes.



Contamination of training data impact

	Contam.	Performance gain est.		
		8B	70B	405B
AGIEval	98	8.5	19.9	16.3
BIG-Bench Hard	95	26.0	36.0	41.0
BoolQ	96	4.0	4.7	3.9
CommonSenseQA	30	0.1	0.8	0.6
DROP	-	-	-	-
GSM8K	41	0.0	0.1	1.3
HellaSwag	85	14.8	14.8	14.3
HumanEval	-	-	-	-
MATH	1	0.0	-0.1	-0.2
MBPP	-	-	-	-
MMLU	-	-	-	-
MMLU-Pro	-	-	-	-
NaturalQuestions	52	1.6	0.9	0.8
OpenBookQA	21	3.0	3.3	2.6
PiQA	55	8.5	7.9	8.1
QuaC	99	2.4	11.0	6.4
RACE	-	-	-	-
SiQA	63	2.0	2.3	2.6
SQuAD	0	0.0	0.0	0.0
Winogrande	6	-0.1	-0.1	-0.2
WorldSense	73	-3.1	-0.4	3.9

Both high

Other Detection methods needed

Seems no impact

Maybe larger n

Discussions