HUMAN & MACHINE INTELLIGENCE | CONVERSATIONAL AI

The Llama 3 Herd of Models

July 23, 2024

Notes on LLama 3 Report

For whatever subarea that you work on, this report is like a gold mine with many details that can be expanded into a research paper.

Statistics

8B, 70B and 405B parameters, context window 128K, trained on **15T** tokens (llama 2 was trained on 1.8T).

16K H100 GPUs for 3.8 * 10^25 FLOPs

Assuming a single H100 delivers 60* 10^12 FLOPs (source), this means that the training requires 120 days.

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 ($\theta = 500,000$)	

SwiGLU:

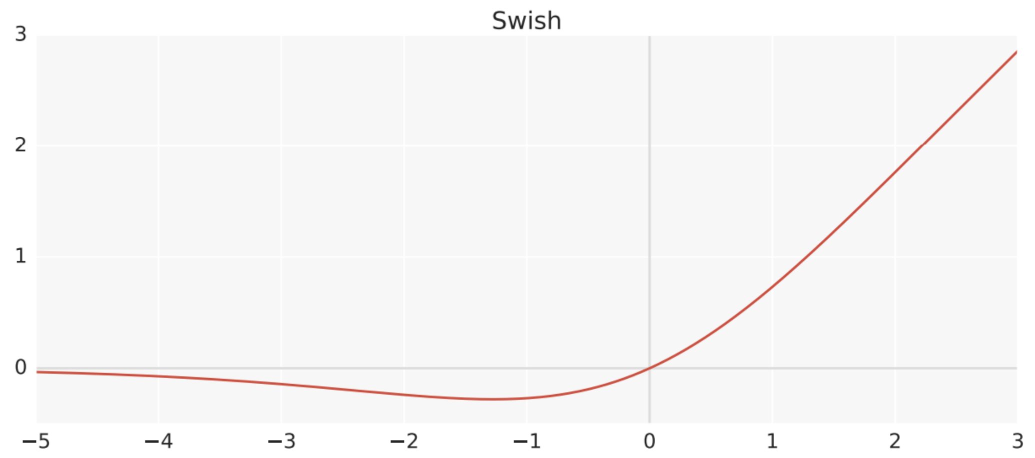


Figure 1: The Swish activation function.

Grouped-Query Attention (Ainslie et al., 2023):

In multi-head attention, each query gets a unique key-value pair. In GQA, some k-v pairs are reused.

Data (Important!!):

Deduplication: The details of deduplication is intricate. See DataComp-LM and FineWeb, which provides detailed ablations on how to perform deduplication.

💡 I don't think there exists a work on understanding the effect of **different deduplication strategies** on memorization: The only work is Deduplicating Training Data Makes Language Models Better (Lee et al., ACL 2022), which shows deduplication can make neural networks memorize less. Furthermore, I don't think there exist a work on understanding what type of data (e.g. code? low-entropy data?) is the model tend to memorize more: Stella Biderman's work only shows that small models tend to memorize a subset of what the large model memorizes.

Heuristic Filtering: Boilerplate text, bad words, outlier words...

Model-based Filtering: FastText, Roberta based text classifiers trained on LLama 2 predictions.

Data Mix: Train several small models on a data mix and use that to predict (similar to what DoReMi does) - is there automated, online ways to do this? See Fan et al., ICML 2024.

Annealing: Upsampling certain data (e.g. Math) significantly improves benchmark scores, and is a very effective way of assessing data quality (Blakeney et al.)

💡 DataComp-LM: In search of the next generation of training sets for language models (Li et al., 2024)

Release an evaluation suite for pre-training dataset curation. Experiment suggests that the most important part is model based filtering.

The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale (Penedo et al., 2024)

Presents a comprehensive study on different heuristic / model based filters and ablations on ways of performing deduplication.

Does your data spark joy? Performance gains from domain upsampling at the end of training (Blakeney et al., COLM 2024)

Discusses the annealing technique at the end of training to assess the quality of smaller datasets.

DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining (Xie et al., NeurIPS 2023)

Scaling Laws:

Establish FLOPs - Downstream Task NLL loss

Establish Downstream task NLL loss - Downstream Acc.

💡 Understanding Emergent Abilities of Language Models from the Loss Perspective (Du et al., 2024)

The pre-training nll loss is loosely related to downstream performance.

Why Has Predicting Downstream Capabilities of Frontier AI Models with Scale Remained Elusive? (Schaeffer et al., 2024)

Suggest to use continuous metrics (the same as what LLama 3 did, nll loss on the downstream task) as opposed to discrete metrics (e.g. Accuracy, F1)

Pre-Training:

Gradually Doubling the Batch Size (4M * 4096 (252M) → 8M * 8192 (2.87T) → 16M * 8192)

The data is not the same throughout training (adding more recent data, adjusting data mix) - however the effect of such adjustments is unclear (no ablations reported).

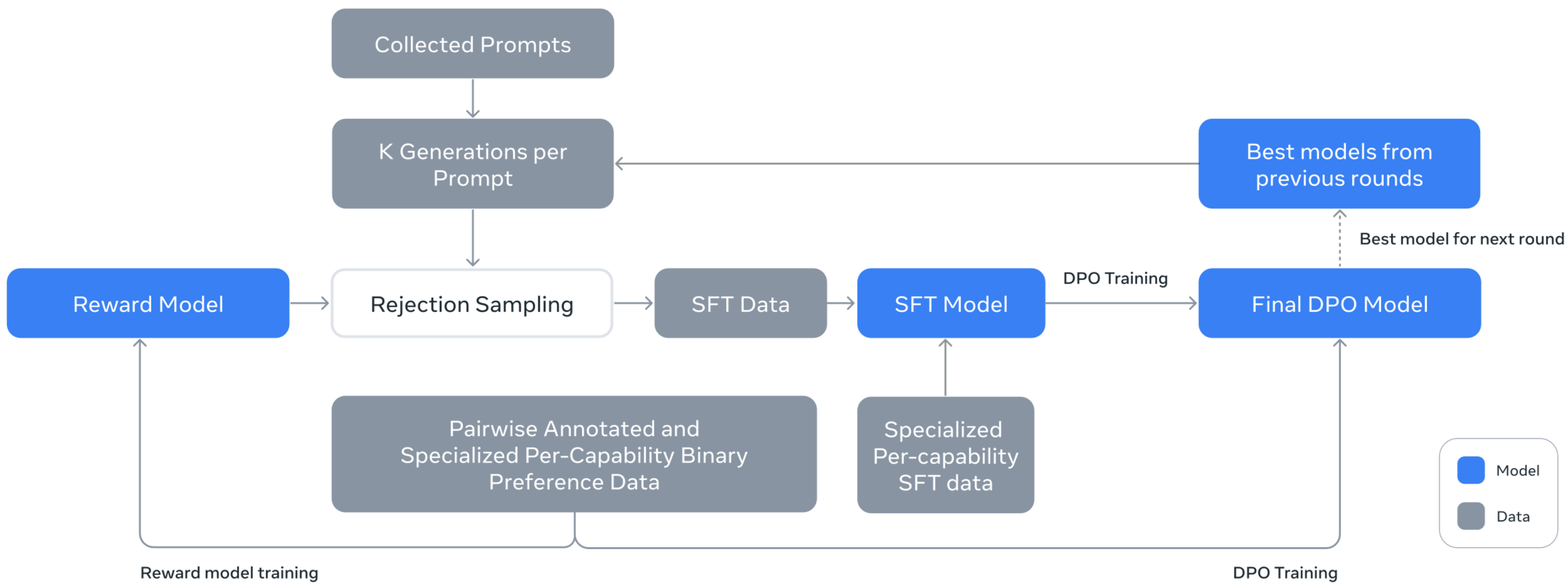
First train on short-context data, then **gradually** increase context length until the model passes the "needle in a haystack" test up to the current length.

Data Engineering for Scaling Language Models to 128K Context (Fu et al., ICML 2024)

Studies how to continue pre-train models to scale from 4K context to 128K. Naive upsampling long-context data yields suboptimal performance, need to have balanced data mixture. (But isn't this different to what Llama 3 does? - maybe the curriculum part is important).

Post-Training (SFT):

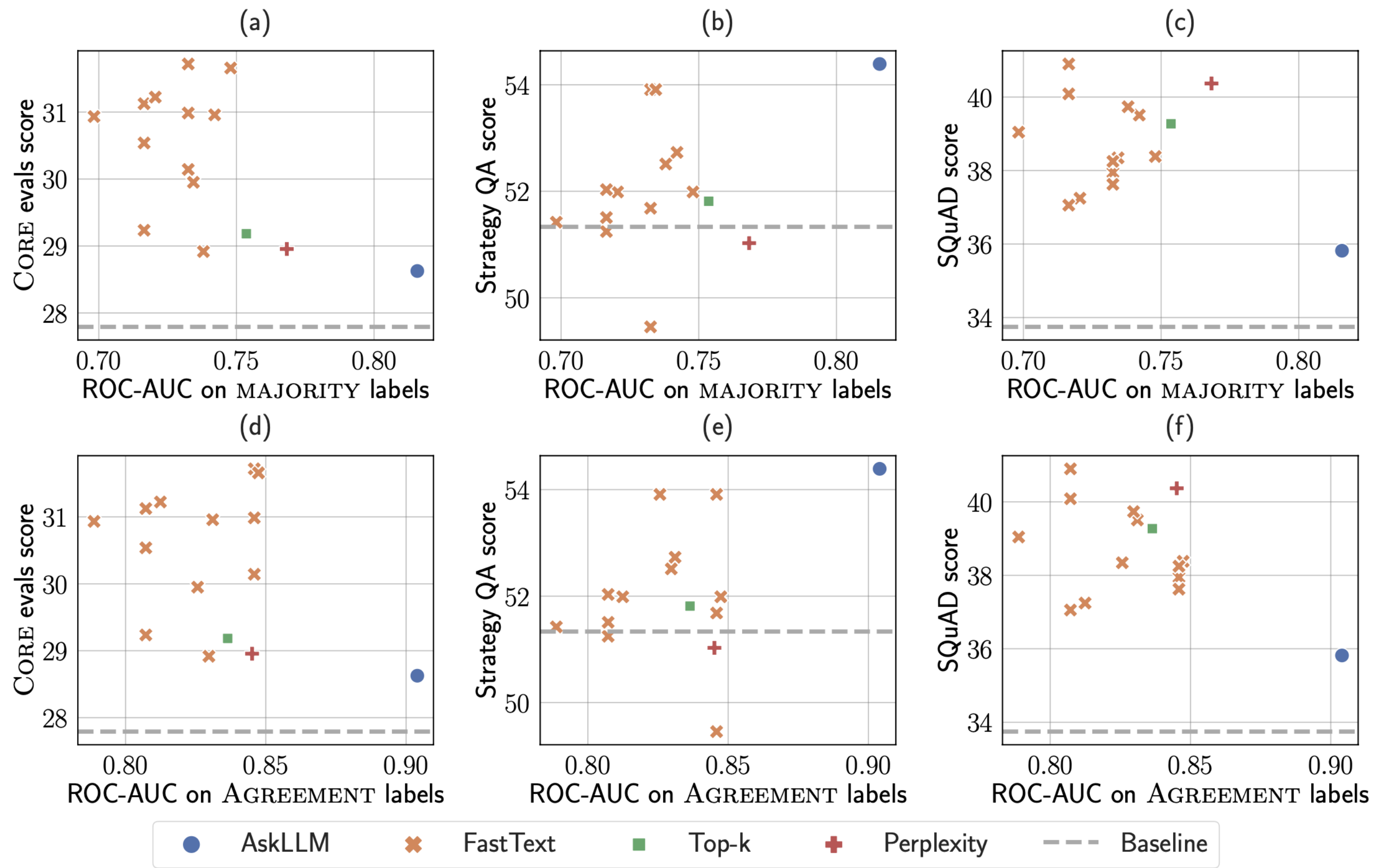
First train reward, sample data from base model, use reward to select best responses and add to SFT mix.



Preference Optimization:

Regularization with NLL loss: Many works have adopted this but really check Haoran's paper who first did this.

Quality Scoring: Reward and model-based scoring have big disagreement. Model-based scoring (prompting) have high correspondence to human judgement (from the DataComp-LM paper, x-axis=agreement with human judgement, y-axis=performance):



Difficulty Scoring: Prompting to get difficulty. Semantic Deduplication (does this help? no ablations are performed).

Smaug: Fixing failure modes of preference optimisation with dpo-positive (Pal et al., 2024)

This paper investigates a common problem of DPO: the likelihood of the preferred response also goes lower as the model trains, since DPO only cares about how much better is the preferred response over the dispreferred one. To mitigate this, the authors added an 0-1 loss on the log prob ratios between the likelihood of the preferred response under the ref model and the likelihood of it under the optimized model: $\max(0, \log \frac{\pi_{\text{ref}}(y|x)}{\pi(y|x)})$.

Self-Rewarding Language Models (Yuan et al., ICML 2024)

Iteratively uses Llama 2 70B to improve itself. Sort of similar to what LLama 3 did, but Llama 3 adopts a more "curriculum" like strategy by gradually adding new capabilities to the model (e.g. in tool use, they first introduced simple tool use then use the model with simple tool use ability to generate synthetic data).

Contrastive Preference Optimization: Pushing the Boundaries of LLM Performance in Machine Translation (Xu et al., ICML 2024)

Reference Free DPO with NLL regularization

Code:

Expert Training - branching the main pre-training run and train on 1T code data (long-context training). Use this and the base model to generate synthetic dialogues.

Iteratively generate data by increasing difficulty, filter the data with execution feedback and intermediate steps.

Easy: Code generation, Hard: Code + Natural Language (Explaining code, fixing bugs, code translation) - Prompt Llama 3 base to get quality annotations

Somehow suggest that the base model, although not able to generate good synthetic data directly, is good as judging the quality of the synthetic data.

Don't Stop Pretraining: Adapt Language Models to Domains and Tasks (Gururangan et al., ACL 2020)

The best way to improve your model on a given domain is to just continue pre-train on that domain (what a surprise).

Magicoder: Empowering Code Generation with OSS-Instruct (Wei et al., ICML 2024)

Multilingual:

Train Expert → Use Expert to Generate Synthetic SFT data (w/ Rej sampling) → Add translated Reasoning data to improve benchmark scores of MGSM

I wonder why reasoning specifically means "Math" in the context of LLM - the reasoning papers mainly target GSM and MATH benchmarks.

Breaking Language Barriers in Multilingual Mathematical Reasoning: Insights and Observations (Chen et al., 2023)

Reasoning:

Overcoming Lack of Prompts: Get math pre-training data and convert to QA format → add stepwise reasoning traces (use llama 3 to verify whether a step by step solution is valid) → train stepwise reward models to filter incorrect reasoning traces (a further enhancement of the previous step) → Add code → Prompt LLama 3 yield correct solutions when encountering incorrect ones.

Though the improvement of each step is not documented, only the final result.

💡 Let's Verify Step by Step (Lightman et al., 2023)
Monte Carlo Tree Search Boosts Reasoning via Iterative Preference Learning (Xie et al., 2024)
ToRA: A Tool-Integrated Reasoning Agent for Mathematical Problem Solving (Gou et al., ICLR 2024)
Improving Reward Models with Synthetic Critiques (Ye et al., 2023)
Critique-out-loud language models (Ankner et al., 2024)

Long Context:

Even when the base model is extended to long context, adding short context SFT data **significantly** impact long-context capabilities.

Add 0.1% long-context data during SFT optimizes perf. between short and long-context tasks. (Though I expect the # of tokens to be much more than 0.1%).

DPO does not affect long-context abilities (Maybe because DPO is a very superficial process: See "The Unlocking Spell" (Lin et al., ICLR 2024))

💡 The Unlocking Spell on Base LLMs: Rethinking Alignment via In-Context Learning (Lin et al., ICLR 2024)
DPO only adjusts the response into a "helpful assistant" tone rather than changing the capability of the LM, and that in-context examples can adjust the SFT LM into the DPO style LM.
Question: given a fixed number of queries, can you determine whether a model has/hasn't gone through a RLHF/DPO process?

Tool Use:

Start by adding simple tools (web search, python interpreter, Wolfram Alpha). A bit difference: this relies more on human annotation.

Curriculum: "To accelerate the annotation process, we start by bootstrapping basic tool use capabilities by finetuning on synthetically generated data from previous Llama 3 checkpoints."

Zero-shot tool use: Prompt LLama 3 to generate queries given document / The Stack.

💡 ReAct: Synergizing Reasoning and Acting in Language Models (Yao et al., ICLR 2023)
API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs (Li et al., EMNLP 2023)
TOOLVERIFIER: Generalization to New Tools via Self-Verification (Mekala et al., 2024)

Factuality

Heavy use of the base model to generate synthetic data.

1. **Extract a data snippet** from the pre-training data.
2. **Generate a factual question** about these snippets (context) by prompting Llama 3.
3. **Sample responses** from Llama 3 to the question.
4. **Score the correctness** of the generations using the original context as a reference and Llama 3 as a judge.
5. **Score the informativeness** of the generations using Llama 3 as a judge.
6. **Generate a refusal** for responses which are consistently informative and incorrect across the generations, using Llama 3.

How effective is this refusal ? If the base model consistently generate incorrect but informative answers, why does this happen? Does it mean that the pre-training data is simply incorrect?

💡 Does fine-tuning llms on new knowledge encourage hallucinations? (Gekhman et al., 2024)

Reducing conversational agents' overconfidence through linguistic calibration (Mielke et al., 2020)