

LLMs for the “GPU-Poor”

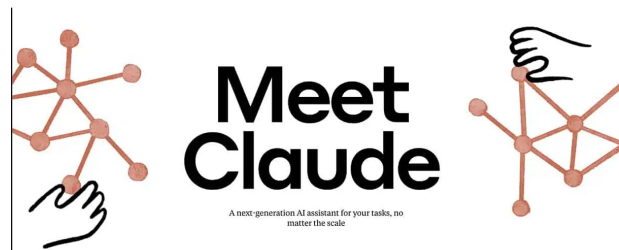
Franck Nijimbere

Founder @GoAgentic

Intro to Large Language Models (LLMs)

An LLM is a type of neural network that specializes in processing, understanding, and generating human language.

Eg. llama-2-70 b



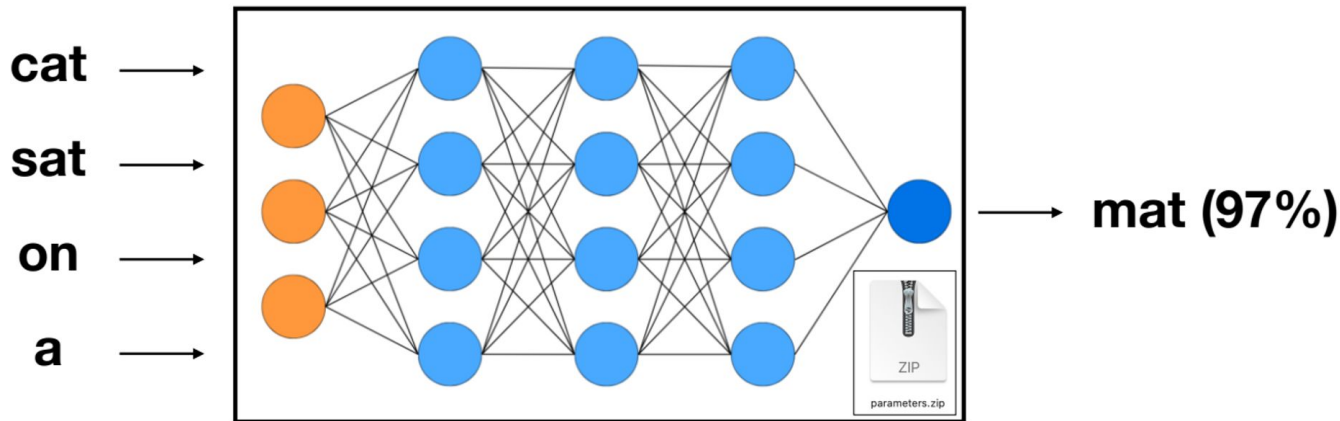
Intro to Large Language Models (LLMs)

There are two main components of an LLM:

1. the **parameters**: model weights
2. the **code** to run the parameters

Given a sequence of words, the LLM predicts the next word.

- This requires parameters/ the network to learn a lot about the world
- All of this info is compressed into the parameters

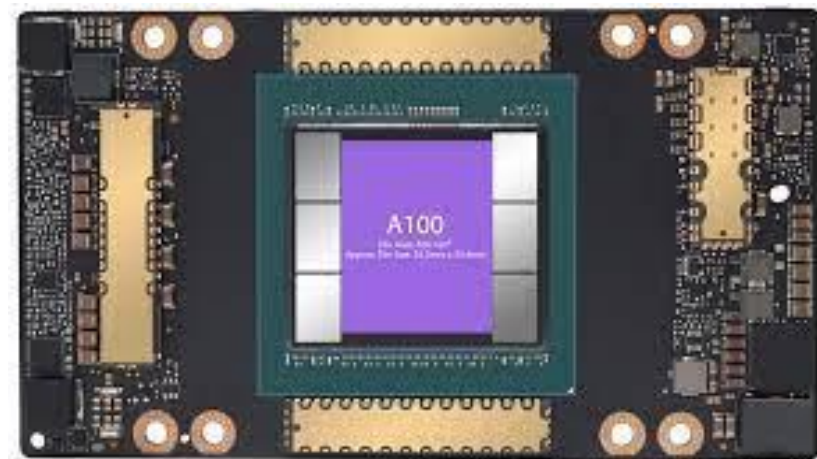


LLM Training: Step1: Model Pre-training

This involves:

1. Taking chunks of the internet collected through crawling (**~10TB of text**)
2. Using a GPU cluster to compute parameters (**6,000 GPUs for 12 days, ~1e24 FLOPS**)
3. Compressing these large amounts of text with lossy compression (**~140 GB file**)

**** values for llama-2-70b**, it's important to note that these values are ~10x worse than current leading models*



To start, we create a base model by learning from a labeled dataset. This process is extremely expensive and usually only done ~1/year.

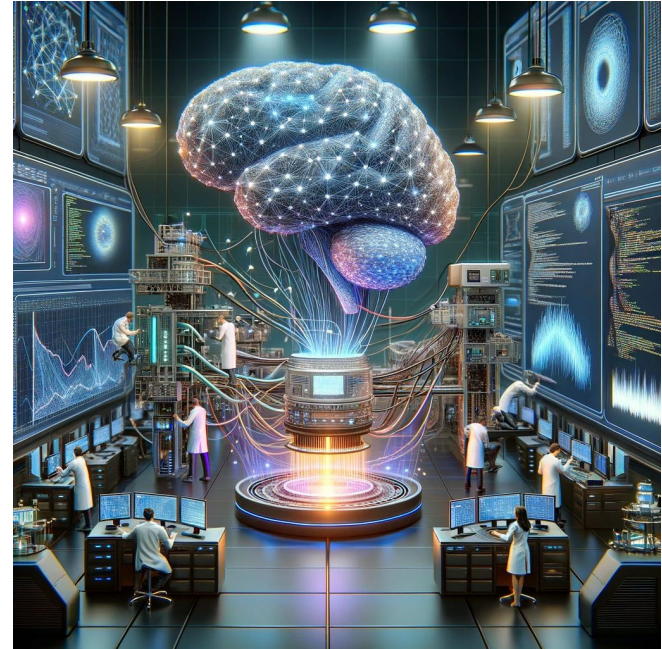
LLM Training: Step2: Model Fine-tuning

This involves:

1. Writing labeling instructions
 1. note: Labeling used to be done by humans. Now, it is increasingly completed by LLMs.
2. Creating a dataset of ~100k high quality specific texts (ex: use ScaleAI)
3. Fine-tuning the base model on this dataset (~1 day)
4. Evaluating and monitoring misbehaviors

E.g Assistant model like ChatGPT

- Fine-tune the model on Q/A `<user: query> + <assistant: answer>` pair formatted data



We take our general base model and tune it to a specific domain and purpose. Unlike pre-training which requires billions of examples, fine-tuning only requires a few hundred domain specific examples.

Model Training vs Model Fine-tuning

	Priority	Data Format
Stage 1	Prioritizes data quantity (requires ~10,000 x more than Stage 2)	general internet documents
Stage 2	Prioritizes data quality and specificity	more specific text format and contents

LLM Training: Step3: Model Inference

How does this work exactly?

The network “dreams” documents. Every-time it generates a new word, this word is fed back in to generate the next word. This allows the network to iteratively build words, and then sentences, and then paragraphs.

However, the model isn't always correct. There is always a risk of hallucinations.

Hallucinations are generated text or responses that are incorrect, fictional, or misleading. This is currently a big problem with LLMs and can be caused by:

1. The LLM was trained on outdated or incorrect information.
2. The LLM was trained on biased or discriminatory training data.
3. The LLM doesn't have access to real-time, real-world information.

LLM scaling laws

We can significantly, and predictably improve LLM performance just by increasing the # of parameters and the amount of training data.

This is very important because:

- It's a very easy and clear path for improvement
- It's why people are rushing to obtain more data and GPUs

GPU cost



NVIDIA A100 ~ \$27K

Market Summary > NVIDIA Corp

475.06 USD

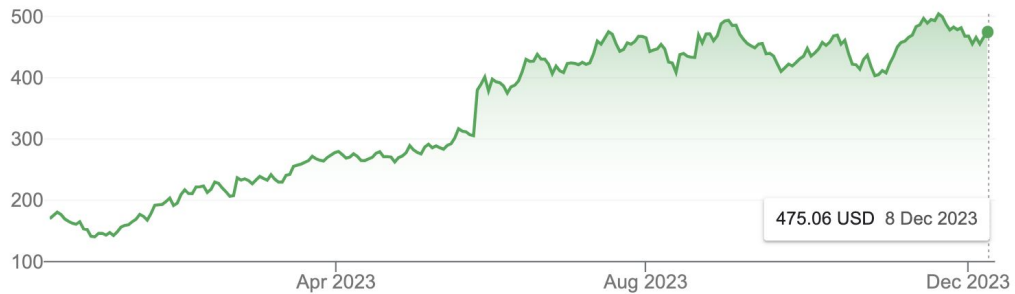
+ Follow

+305.05 (179.43%) ↑ past year

Closed: Dec 8, 19:59 EST • Disclaimer

After hours 474.97 -0.090 (0.019%)

1D | 5D | 1M | 6M | YTD | 1Y | 5Y | Max



GPU-Poor



+ Subscribe

Sign in

The GPU-Poor

Then there are a whole host of startups and open-source researchers who are struggling with far fewer GPUs. They are spending significant time and effort attempting to do things that simply don't help, or frankly, matter. For example, many researchers are spending countless hours agonizing on fine-tuning models with GPUs that don't have enough VRAM. This is an extremely counter-productive use of their skills and time.



Joey (e/λ)
@shxf0072

Being GPU-Poor is blessing not curse



hardmaru ✓
@hardmaru

I prefer to operate in “GPU-Poor” mode.

I don't agree with the take from the semianalysis piece. Creative breakthroughs often occur under constraints—new systems, models, and methods that can better take advantage of even larger-scale compute



Loreto Parisi 🇺🇦 🇪🇺
@loretoparis

In fact thanks to "the Poor GPU" we had FlashAttention, AliBi, PI, Scaled RoPE, bf16, int8, int4, GPTQ, LoRA, QLoRA, and again ReRoPE, RecycleGPT, and of course Llama.cpp, Whisper.cpp and the whole (and brand new) C++ #LLM and DL frameworks waves. 🏄. Missed something?



Nabarun ✓ @nbrsr · Aug 28

GPU-poor, that will be almost 98% of all academic researcher who had



Manu Romero ✓ @mrm8488 · Aug 29

Thanks "GPU-Poor" guys for the optimization/quantiz



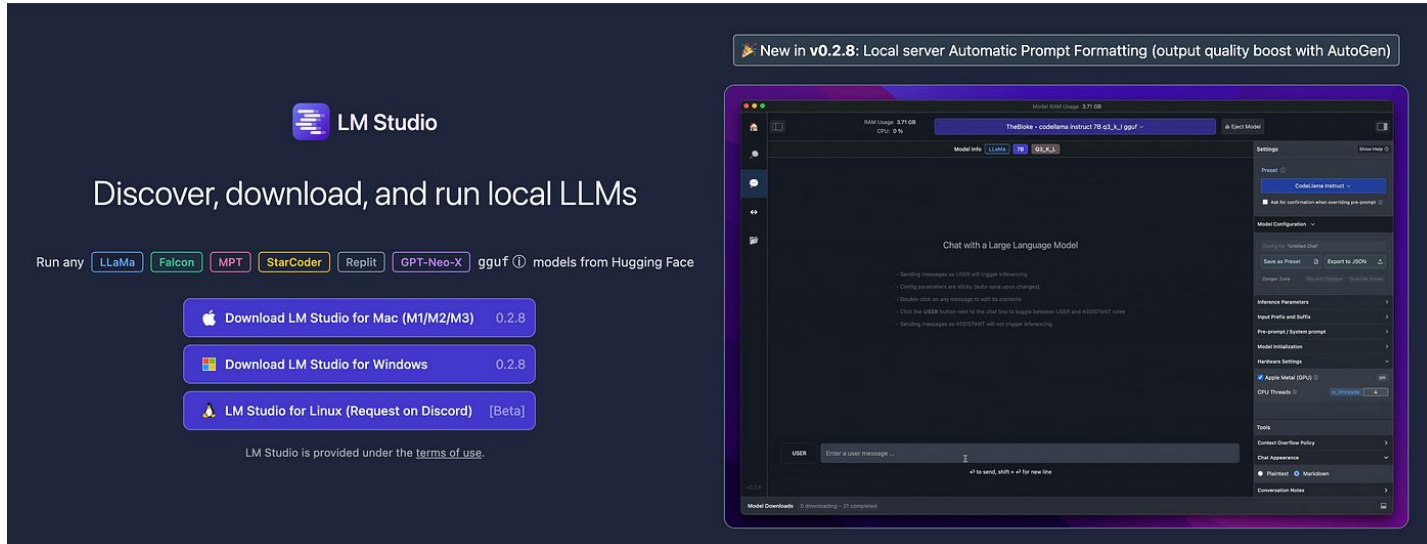
Dimitris Papailiopoulos ✓ @DimitrisPapail · A

We are not **GPU-poor**, we are CPU-rich!

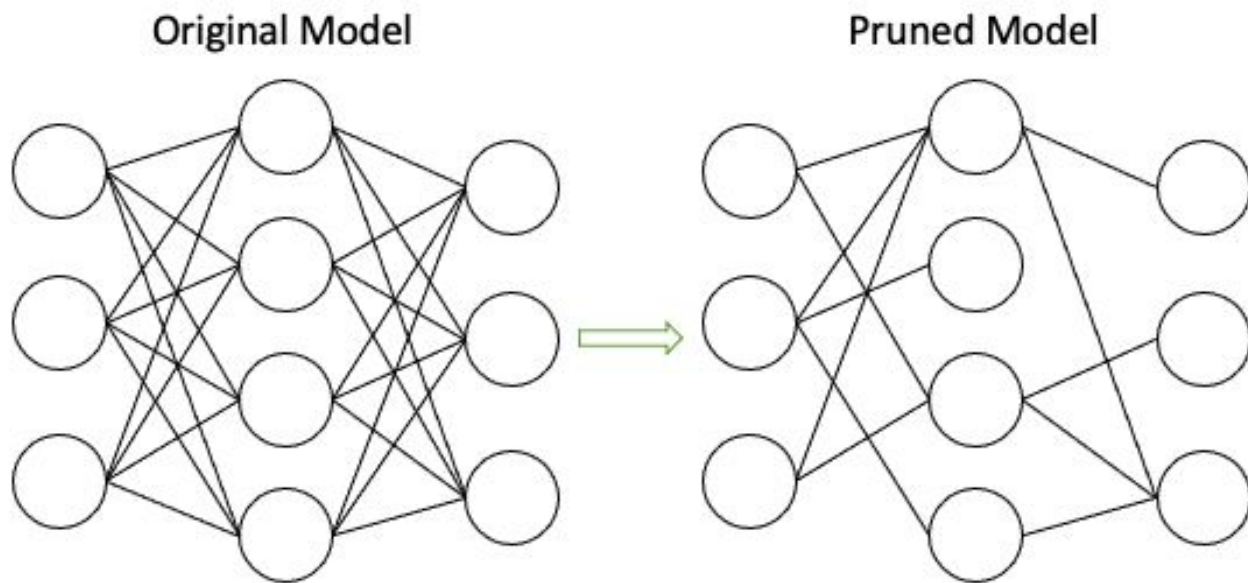
On-device inference

The logo for LLaMA++ features the text "LLaMA" in white and "++" in orange on a black background.

The main goal of llama.cpp is to run the LLaMA model using 4-bit integer quantization on a MacBook

A screenshot of the LM Studio application interface. The top banner reads "New in v0.2.8: Local server Automatic Prompt Formatting (output quality boost with AutoGen)". The main area is titled "LM Studio" and "Discover, download, and run local LLMs". It lists various models from Hugging Face: LLaMa, Falcon, MPT, StarCoder, Replit, and GPT-Neo-X. Below this are three download buttons: "Download LM Studio for Mac (M1/M2/M3) 0.2.8", "Download LM Studio for Windows 0.2.8", and "LM Studio for Linux (Request on Discord) [Beta]". At the bottom, it states "LM Studio is provided under the terms of use." On the right, a preview window shows the chat interface with a model selected as "TheBloke - codellama Instruct 7B G3_KJ gguf". The chat area contains instructions on how to use the interface, such as "Sending messages as USER will trigger inferencing" and "Enter a user message" in the input field. The right sidebar shows settings for the selected model, including inference parameters and hardware settings.

LLM Optimization: Model Pruning



For example, SparseGPT claims their algorithm can prune models by 50% without retraining.

LLM Optimization: Model Quantization

Quantization (post-training) — Normalize and round the weights. No retraining is needed.

Mixed precision — Using a combination of lower (e.g., float16) and higher (e.g., float32) precision arithmetic to balance performance and accuracy.

Quantization

Floating point

3452.3194



Integer

3452

32 bit



8 bit



LLM Optimization: Low-rank factorization

LoRa (Low-Rank Adaptation of Large Language Models) — A method to reduce the model size and computational requirements by approximating large matrices using low-rank decomposition.

[Submitted on 17 Jun 2021 (v1), last revised 16 Oct 2021 (this version, v2)]

LoRA: Low-Rank Adaptation of Large Language Models

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen

[Submitted on 5 Sep 2023]

Delta-LoRA: Fine-Tuning High-Rank Parameters with the Delta of Low-Rank Matrices

Bojia Zi, Xianbiao Qi, Lingzhi Wang, Jianan Wang, Kam-Fai Wong, Lei Zhang

[Submitted on 26 Sep 2023 (v1), last revised 9 Oct 2023 (this version, v2)]

QA-LoRA: Quantization-Aware Low-Rank Adaptation of Large Language Models

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, Xiaopeng Zhang, Qi Tian

[Submitted on 6 Nov 2023 (v1), last revised 7 Nov 2023 (this version, v2)]

S-LoRA: Serving Thousands of Concurrent LoRA Adapters

Ying Sheng, Shiyi Cao, Dacheng Li, Coleman Hooper, Nicholas Lee, Shuo Yang, Christopher Chou, Banghua Zhu, Lianmin Zheng, Kurt Keutzer, Joseph E. Gonzalez, Ion Stoica

[Submitted on 23 May 2023]

QLoRA: Efficient Finetuning of Quantized LLMs

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, Luke Zettlemoyer

[Submitted on 21 Sep 2023]

LongLoRA: Efficient Fine-tuning of Long-Context Large Language Models

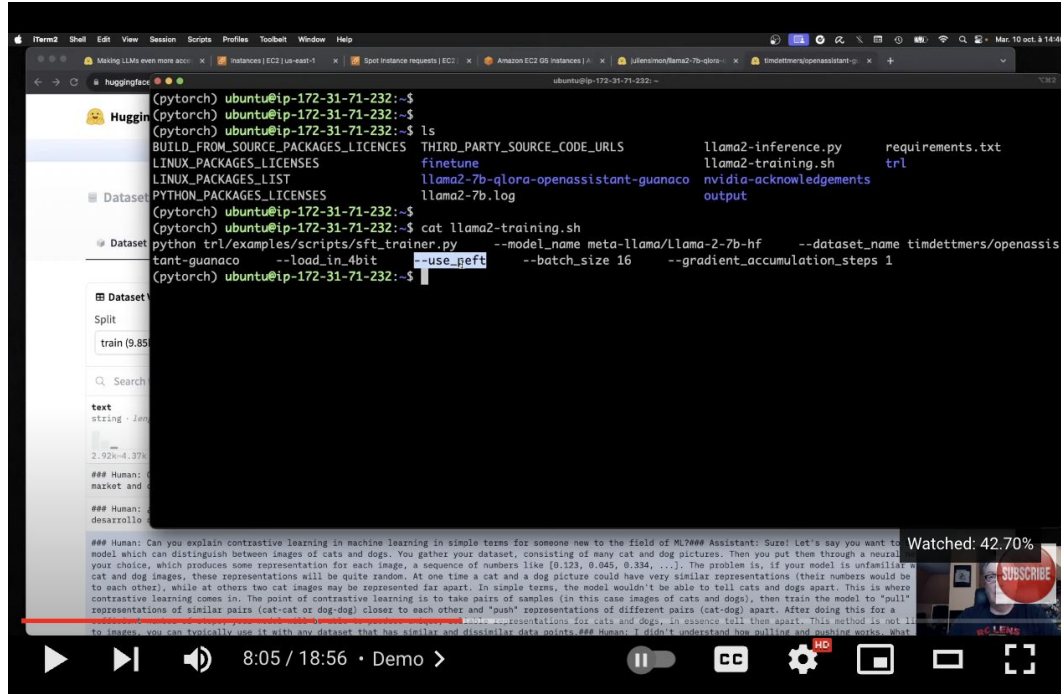
Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, Jiaya Jia

[Submitted on 12 Oct 2023 (v1), last revised 28 Nov 2023 (this version, v4)]

LoftQ: LoRA-Fine-Tuning-Aware Quantization for Large Language Models

Yixiao Li, Yifan Yu, Chen Liang, Pengcheng He, Nikos Karampatziakis, Weizhu Chen, Tuo Zhao

Fine-tune LLMs for a few dollars



A fistful of dollars: fine-tune LLaMA 2 7B with QLoRA



Julien Simon
9.4K subscribers

Subscribe

38



Share



Final words

Lots of opportunities for the GPU-Poor:

- Restrict the domain (fine-tune models): model for kwatura, generating Burundian proverbs, generating ibisokozo
- Dispatch multiple smaller models, each specialised in a sub-task, to improve overall performance: GoAgentic.
- Improve optimization techniques for the GPU-poor.