The LLM Scientist Roadmap

A Complete Free Pack of Resources

1. LLM Architecture
2. PreTraining Models
3. Post Training Dataset
4. Supervised FineTune
5. Preference Alignment
6. Evaluation
7. Quantization
8. New Trends

The LLM Architecture

You don't need to be a Transformer whiz—just grasp these core steps:

- **Model evolution**: From encoder–decoder setups to decoder-only (e.g. GPT).
- **Tokenization**: Splitting text into tokens and turning them into numbers.
- **Attention**: Using (self-)attention layers to track context across long passages.
- **Sampling**: Generating text via greedy/beam search or probabilistic methods like temperature and nucleus sampling.

Free Resources

- Introduction to Transformer by Analytics Vidhya
- Visual intro to Transformers by 3Blue1Brown: Visual introduction to Transformers for complete beginners.
- LLM Visualization by Brendan Bycroft: Interactive 3D visualization of LLM internals.
- nanoGPT by Andrej Karpathy: A 2h-long YouTube video to reimplement GPT from scratch (for programmers). He also made a video about tokenization.

Pre-training Models

Pre-training is resource-heavy but useful to understand. Hobbyists can still try it with smaller (<1B) models.

- **Data**: Massive, cleaned datasets (e.g. 15T tokens for Llama 3.1) are required.
- **Training**: Combines data, pipeline, and tensor parallelism—needs fast GPU coordination.
- **Optimization**: Uses tricks like warm-ups, gradient clipping, mixed precision, and modern optimizers (AdamW, Lion).
- **Monitoring**: Track loss, gradients, GPU usage via dashboards and profiling tools.

Free Resources

- FineWeb by Penedo et al.: Article to recreate a large-scale dataset for LLM pretraining (15T), including FineWeb-Edu, a high-quality subset.
- **RedPajama v2** by Weber et al.: Another article and paper about a large-scale pre-training dataset with a lot of interesting quality filters.
- nanotron by Hugging Face: Minimalistic LLM training codebase used to make **SmolLM2**.

Post Training Dataset

Post-training data includes instruction–answer pairs (for fine-tuning or alignment). Since chat-style data is rare, we refine seed data to improve quality and variety.

- **Formats**: Stored in ShareGPT/OpenAI style, then converted to templates like ChatML or Alpaca.
- **Synthetic Data**: Use models (e.g., GPT-4o) to generate instruction–response pairs from seed tasks.
- **Enhancement**: Improve data with tested outputs, multiple answers, Chain-of-Thought, personas, etc.
- **Filtering**: Clean data using rules, deduplication (e.g., MinHash), and quality checks with reward models or LLM judges.

Free Resources

- Synthetic Data Generator by Argilla: Beginner-friendly way of building datasets using natural language in a Hugging Face space.
- **LLM Datasets** by Maxime Labonne: Curated list of datasets and tools for post-training.
- NeMo-Curator by Nvidia: Dataset preparation and curation framework for pre- and post-training data.

Supervised FineTuning

SFT teaches models to follow instructions by refining existing knowledge. Focus on data quality over hyperparameter tuning.

- **Methods**: Full fine-tuning (high compute) vs. LoRA/QLoRA (memory-efficient adapters).
- **Tools**: TRL, Unsloth, Axolotl.
- **Key Params**: LR, batch size, epochs, optimizer, warmup, plus LoRA's rank, alpha, target modules.
- **Scaling**: Use DeepSpeed or FSDP with ZeRO & checkpointing for GPU efficiency.
- **Monitoring**: Track loss, LR, gradients; watch for spikes or instability.

Free Resources

- Fine-tune Llama 3.1 Ultra-Efficiently with Unsloth by Maxime Labonne: Hands-on tutorial on how to fine-tune a Llama 3.1 model using Unsloth.
- Axolotl - Documentation by Wing Lian: Lots of interesting information related to distributed training and dataset formats.
- Mastering LLMs by Hamel Husain: Collection of educational resources about fine-tuning (but also RAG, evaluation, applications, and prompt engineering).

Preference Alignment

This stage tunes model responses to match human preferences—reducing toxicity, hallucinations, and improving usefulness. Key methods: DPO, GRPO, and PPO.

- **Rejection Sampling**: Generate and score multiple replies; pick the best for training.

- **DPO**: Efficiently favors better responses without a reward model—ideal for chat.

- **Reward Models**: Use human feedback to score outputs. Tools: TRL, verl, OpenRLHF.

- **RL (GRPO, PPO)**: Reinforcement learning boosts quality via rewards—powerful but compute-heavy.

Free Resources

- Illustrating RLHF by Hugging Face: Introduction to RLHF with reward model training and fine-tuning with reinforcement learning.
- LLM Training: RLHF and Its Alternatives by Sebastian Raschka: Overview of the RLHF process and alternatives like RLAIF.
- Preference Tuning LLMs by Hugging Face: Comparison of the DPO, IPO, and KTO algorithms to perform preference alignment.

Evaluation

Essential for improving models, but tricky—beware Goodhart's Law.

- **Benchmarks**: Fast and objective, but weak on creativity and prone to data leaks.
- **Human Eval**: Best for tone and nuance, less reliable for facts.
- **Model Judges**: Scalable, matches human taste, but can be biased.
- **Feedback**: Use errors to guide better data and training.

Free Resources

- Evaluation guidebook by Clémentine Fourrier: Practical insights and theoretical knowledge about LLM evaluation.
- Open LLM Leaderboard by Hugging Face: Main leaderboard to compare LLMs in an open and reproducible way (automated benchmarks).
- Language Model Evaluation Harness by EleutherAI: A popular framework for evaluating LLMs using automated benchmarks.
- Lighteval by Hugging Face: Alternative evaluation framework that also includes model-based evaluations.
- Chatbot Arena by LMSYS: Elo rating of general-purpose LLMs, based on comparisons made by humans (human evaluation).

Quantization

Quantization reduces model size and compute by using lower precision (e.g., 4-bit instead of 16-bit weights).

- **Basics**: Learn precision types (FP32, FP16, INT8) and simple methods like absmax or zero-point scaling.

- **Tools**: llama.cpp + GGUF format make running LLMs on CPUs easy and efficient.

- **Advanced Methods**:
  - GPTQ/EXL2, AWQ: Use per-layer calibration to keep quality at low bit sizes.
  - SmoothQuant, ZeroQuant: Preprocess and optimize models to handle outliers and reduce hardware load.

Free Resources

- Introduction to quantization by Maxime Labonne: Overview of quantization, absmax and zero-point quantization, and LLM.int8() with code.
- Quantize Llama models with llama.cpp by Maxime Labonne: Tutorial on how to quantize a Llama 2 model using llama.cpp and the GGUF format.
- 4-bit LLM Quantization with GPTQ by Maxime Labonne: Tutorial on how to quantize an LLM using the GPTQ algorithm with AutoGPTQ.
- Understanding Activation-Aware Weight Quantization by FriendliAI: Overview of the AWQ technique and its benefits.

Quantization

Some advanced topics don't fit neatly elsewhere—ranging from proven techniques to cutting-edge research.

- **Model Merging**: Combine models without fine-tuning (e.g., SLERP via mergekit).
- **Multimodal**: Models like CLIP or LLaVA handle text, images, and more.
- **Interpretability**: Tools like SAEs and abliteration reveal or tweak model behavior.
- **Test-Time Compute**: Improve reasoning by using more compute during inference (e.g., PRMs, MCTS).

Free Resources

- Merge LLMs with mergekit by Maxime Labonne: Tutorial about model merging using mergekit.
- Smol Vision by Merve Noyan: Collection of notebooks and scripts dedicated to small multimodal models.
- Large Multimodal Models by Chip Huyen: Overview of multimodal systems and the recent history of this field.
- Unsensor any LLM with abliteration by Maxime Labonne: Direct application of interpretability techniques to modify the style of a model.
- Intuitive Explanation of SAEs by Adam Karvonen: Article about how SAEs work and why they make sense for interpretability.