SAMIR GENAI ACADEMY

PROFESSIONAL STUDY GUIDE SERIES

Phase 01 · GenAI Foundations

05

Fine-Tuning & PEFT

LoRA, QLoRA, instruction tuning — customise any model

Module 05 of 25 · **1 hour**

Instructor: **Samir Kumar** · Edition 2026

STUDY GUIDE

01 Overview

LoRA, QLoRA, instruction tuning — customise any model. This guide gives you the core theory, the tools that matter, a hands-on build, working code, and the pitfalls to avoid — everything you need to learn the topic to a professional standard and apply it in real projects.

1 hour	4 + 4	5	05/25
LESSON LENGTH	CONCEPTS + BUILD STEPS	FRAMEWORKS & TOOLS	PHASE 01

02 Learning Objectives

- Decide when fine-tuning beats prompting or RAG.
- Explain LoRA and QLoRA parameter-efficient fine-tuning.
- Prepare an instruction-tuning dataset.
- Fine-tune an open model on a consumer GPU / Colab.

03 Core Concepts

When to fine-tune

Prompting and RAG solve most problems. Fine-tune when you need a consistent style/format, a narrow domain behaviour, lower latency/cost via a smaller model, or to teach a skill that prompts can't reliably elicit. Fine-tuning teaches behaviour, not facts — use RAG for facts.

PEFT: LoRA & QLoRA

Full fine-tuning updates all weights — expensive. LoRA freezes the base model and trains small low-rank adapter matrices, cutting trainable parameters by orders of magnitude. QLoRA adds 4-bit quantization so you can fine-tune large models on a single consumer GPU.

Dataset quality

Results live or die on data. Instruction tuning needs clean (prompt, ideal-response) pairs that reflect production inputs. A few hundred high-quality, diverse examples beat tens of thousands of noisy ones.

Evaluation & deployment

Hold out a test set, compare against the base model with prompting, and watch for catastrophic forgetting. Adapters are tiny and can be swapped or merged at deploy time.

04 Visual Diagram

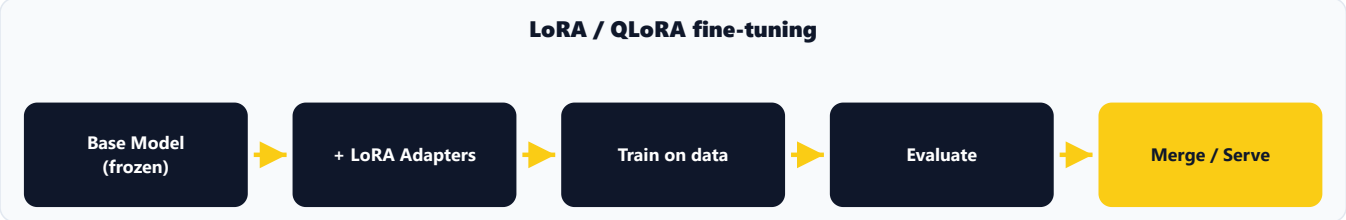

Figure 5 — Only the small low-rank adapter matrices are trained; the large base weights stay frozen, making fine-tuning cheap.

05 Formula Spotlight

Low-Rank Adaptation (LoRA)

$$W' = W_0 + \Delta W = W_0 + B \cdot A$$

W_0	frozen pretrained weight matrix (d × k)
B, A	trainable low-rank matrices, B is d × r and A is r × k
r	rank, with r ≪ min(d, k) — e.g. 8 or 16
$\Delta W = B \cdot A$	the learned update, expressed with far fewer parameters

Instead of updating all d×k weights, LoRA learns two skinny matrices whose product approximates the needed change. Trainable parameters drop by ~99%, so a 7B model fine-tunes on a single consumer GPU (QLoRA adds 4-bit quantization on top).

06 Key Frameworks & Tools

Hugging Face PEFT · bitsandbytes (QLoRA) · TRL (SFTTrainer) · Unsloth · Google Colab

07 Hands-On Build

🛠 LoRA Fine-tuned Model on Consumer GPU / Colab

1. Curate 200-500 high-quality instruction pairs for a narrow task.
2. Load a 7B open model in 4-bit (QLoRA) on Colab.
3. Train LoRA adapters with SFTTrainer for a few epochs.
4. Evaluate against the base model on a held-out set; merge or save adapters.

08 Code Walkthrough

LoRA config with Hugging Face PEFT — PYTHON

```
from peft import LoraConfig, get_peft_model

config = LoraConfig(
    r=16, lora_alpha=32, lora_dropout=0.05,
    target_modules=["q_proj", "k_proj", "v_proj", "o_proj"],
    task_type="CAUSAL_LM",
)
model = get_peft_model(base_model, config)
model.print_trainable_parameters()  # ~0.2% of params trainable
```

09 💉 Lab Assistant

Lab 5 · QLoRA fine-tune on Colab

Prerequisites: Colab GPU runtime · pip install peft trl bitsandbytes transformers

1. **Load a 7B model in 4-bit and attach a LoRA config (r=16)**
   > Expected: print_trainable_parameters shows ~0.1% trainable
2. **Train SFTTrainer for 1-3 epochs on 200-500 pairs**
   > Expected: Loss decreases steadily
3. **Compare tuned vs base on a held-out set**
   > Expected: Tuned model matches your target style/format better

Troubleshooting

- CUDA OOM → lower batch size or sequence length.
- No improvement → data too small/repetitive; diversify it.

10 Pitfalls & Key Takeaways

⚠ COMMON PITFALLS

- Fine-tuning to inject facts — use RAG instead.
- Training on small, repetitive data and overfitting.
- Skipping the base-model baseline, so you can't prove the tune helped.

✓ KEY TAKEAWAYS

- Fine-tune for behaviour/style; use RAG for knowledge.
- LoRA/QLoRA make customisation cheap and laptop-friendly.
- Data quality dominates outcomes — curate ruthlessly.

📺 **Companion video:** Watch the module 05 lesson in your Student Dashboard, then use this guide to revise, copy the code, and complete the build. Download both for offline study.

11 Further Resources

QLoRA paper (Dettmers et al.) • Hugging Face PEFT & TRL docs • Unsloth notebooks