Fine-Tuning LLMs

A two-part blueprint for AI builders

@shivanivirdi

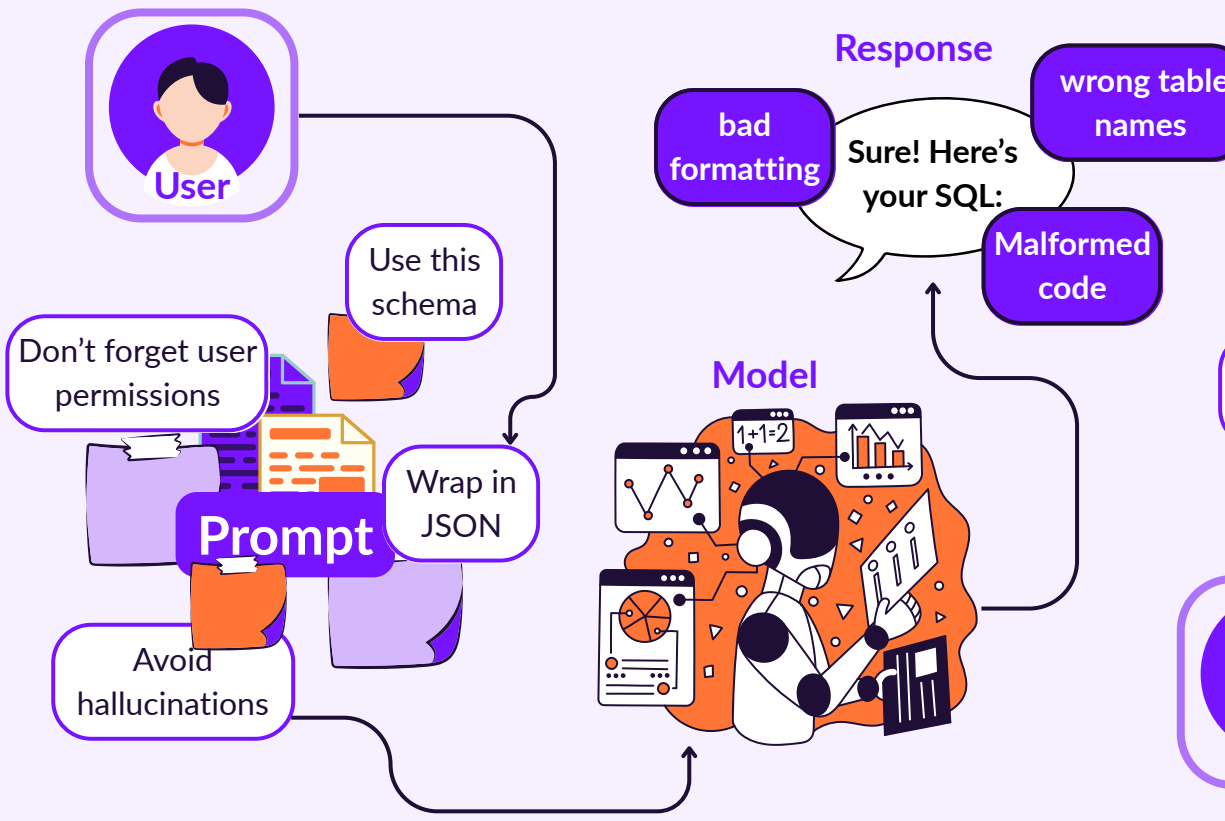
What is Fine-Tuning?

Fine-tuning = updating a **model's internal weights** using new data.

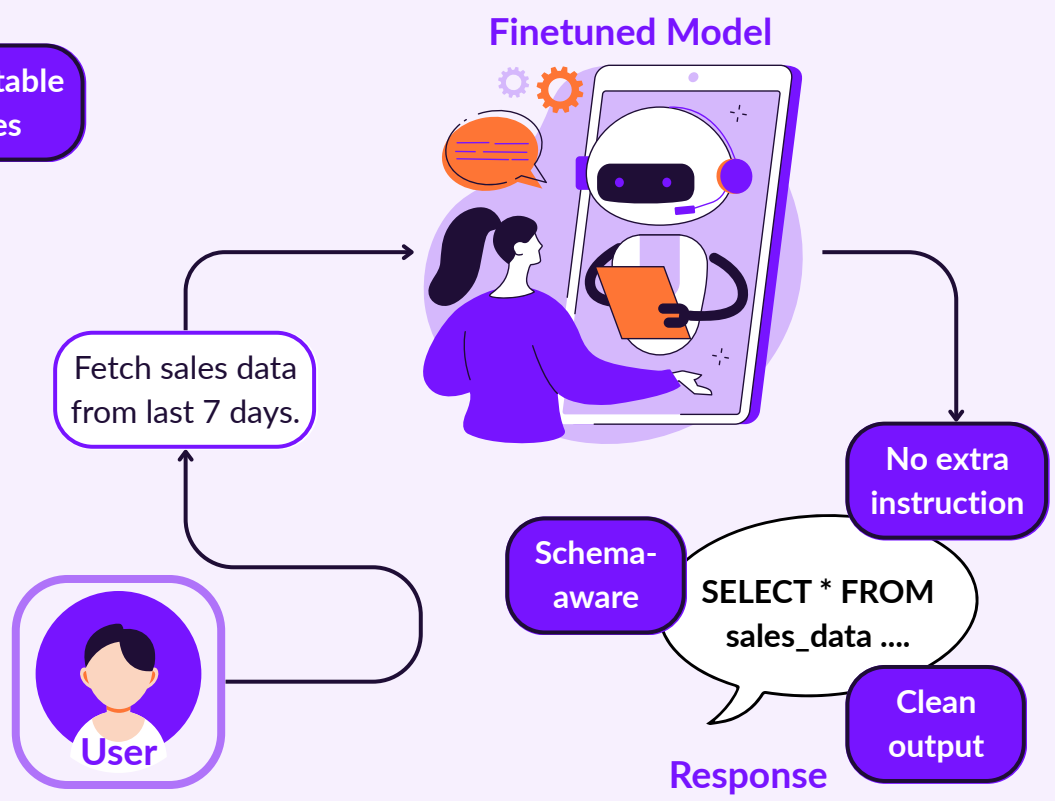
Unlike prompting or retrieval, which guide the model from the outside, **Fine-tuning changes how the model generalizes.**

It's about rewriting behaviour at the model level.

When Guidance Isn't Enough



Prompting steers the output but the model still doesn't understand the task.



Fine-tuning bakes the behaviour into the model, no hacks required.

@shivanivirdi

NEOSAGE

Why This Matters

LLM fine-tuning is one of the most misunderstood and overused tools in AI.

Used right, it creates models that reliably follow structure, tone, and domain logic.

Used incorrectly, it silently breaks safety, generalization, and cost.

This guide is what I wish I had when I was starting out.

@shivanivirdi

Part 1: The Strategic Layer

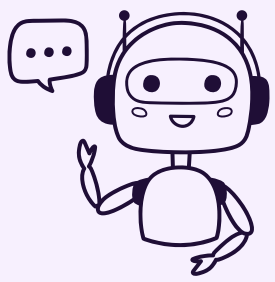
Before fine-tuning, you need to answer two questions:

- Is the problem with knowledge or behaviour?
- Will prompting or RAG solve it faster?

Fine-tuning is what you reach for when the model's behaviour needs to change.

Where Fine-Tuning Actually Sits

Pretraining



Base model

Learns general language patterns from billions of tokens
You don't control this.

Instruction Tuning

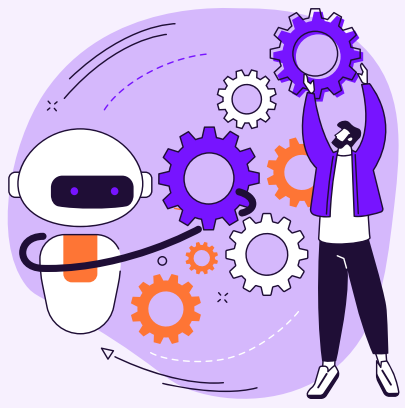
Learns to follow prompts using supervised Q&A, reasoning, formatting
Still foundation model territory.

Alignment

Aligned via RLHF/DPO to be helpful, harmless, honest
Trained on broad, general-purpose preferences.

Your Fine-Tuning

Modifies the model after alignment
You adapt it to your domain, your format, your constraints.



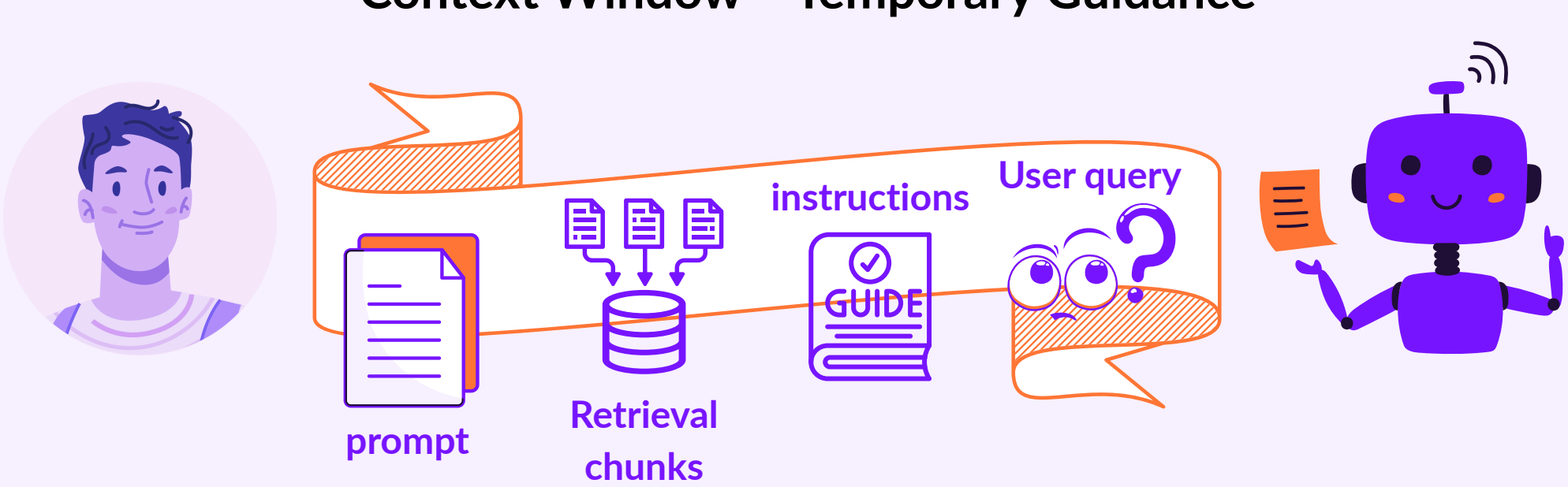
@shivanivirdi

NEOSAGE

Context vs. Weights

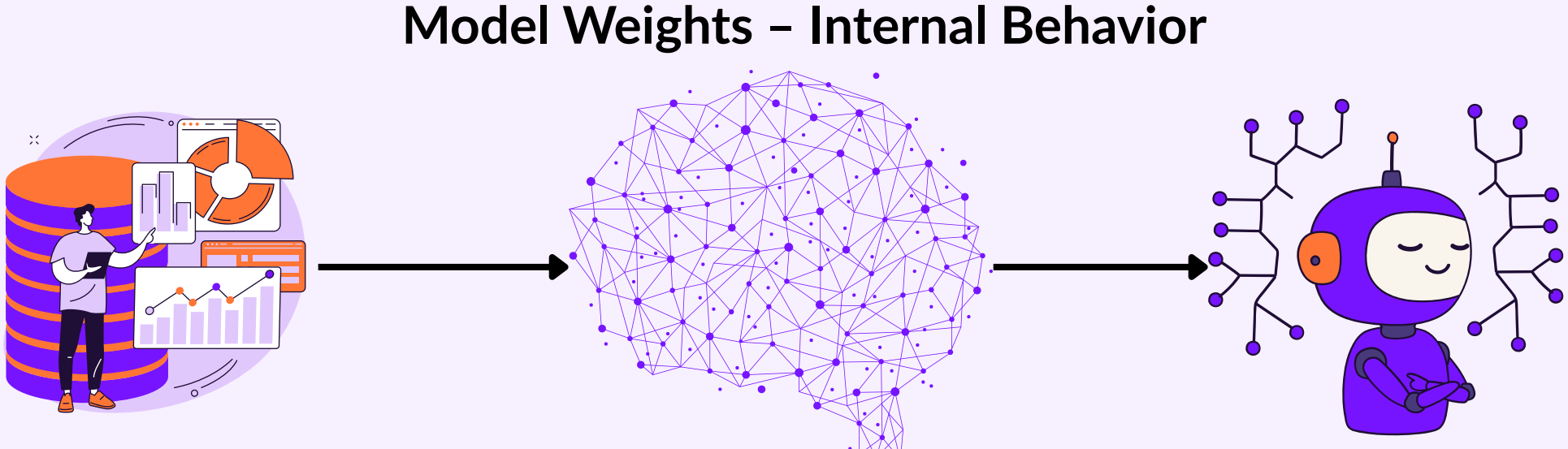
What You Inject vs. What the Model Remembers

Context Window – Temporary Guidance



Prompting & Retrieval operate here
Info must be repeated every time
No real learning happens

Model Weights – Internal Behavior



Fine-tuning updates this
Behaviour persists even without context
Model starts to “know” your logic

@shivanivirdi

NEOSAGE

When Fine-Tuning Makes Sense

Use it when:

- You need a strict structure (e.g. always emit JSON)
- Tasks require nuanced reasoning
- You're operating in low-resource domains (e.g. medical, legal)
- You want SOTA large model like behaviour in a cheaper model

@shivanivirdi

NEOSAGE

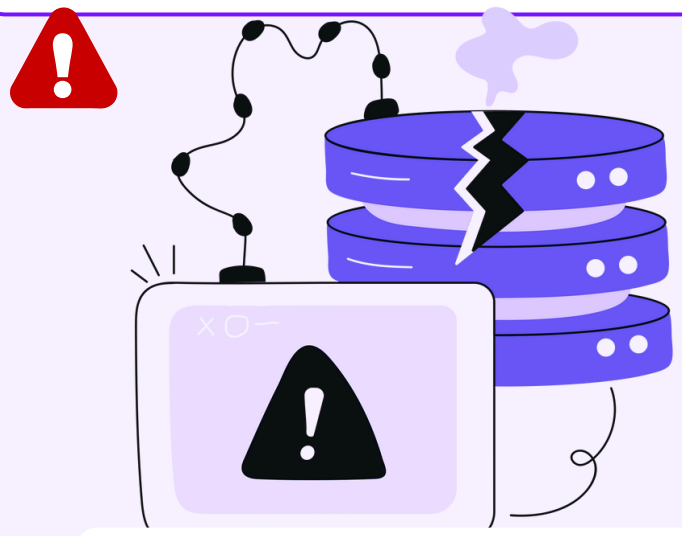
When to Avoid It

Fine-Tuning: 4 Red Flags

Before you commit, make sure you're not trying to solve the wrong problem.

Insufficient High-Quality Data

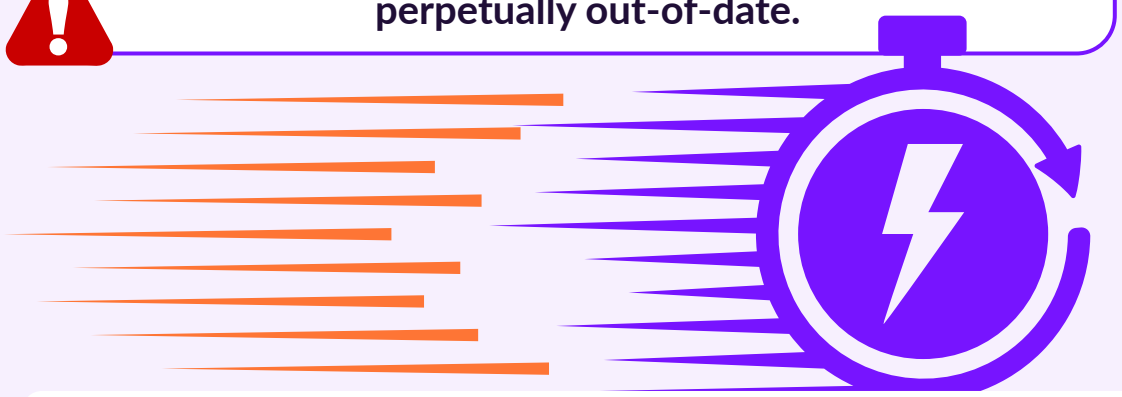
The Problem: The model will overfit or learn incorrect patterns from a small, noisy, or inconsistent dataset. "Garbage in, garbage out."



Validate the task with few-shot prompting (ICL) first. If that doesn't work, fine-tuning likely won't either.

Volatile, Fast-Changing Information

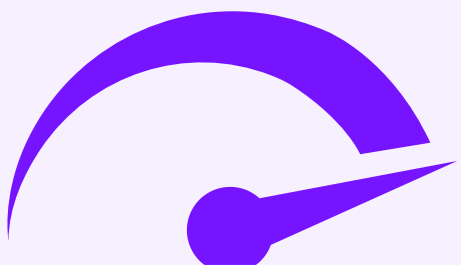
The Problem: Fine-tuning teaches persistent skills, not knowledge that changes daily. The model will be perpetually out-of-date.



Use Retrieval-Augmented Generation (RAG) to provide fresh, real-time information at the moment of the query.

Strict Deployment Constraints

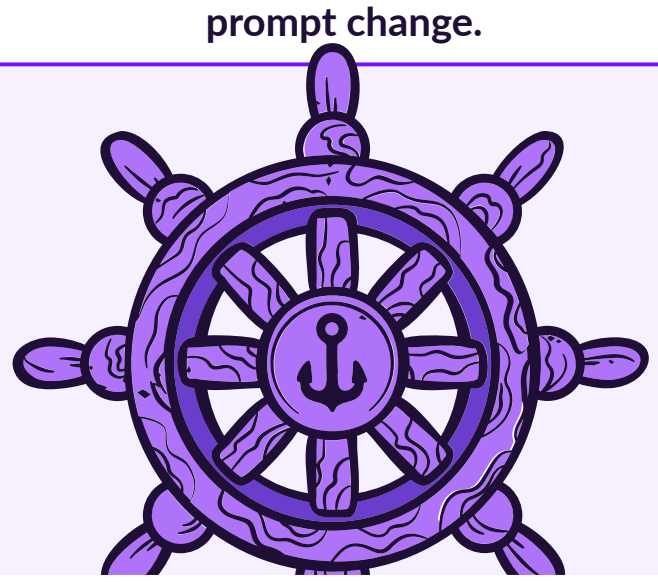
Fine-tuned models are large and can be slow, unsuitable for on edge applications with tight latency or memory budgets.



Use a smaller base model, an optimized API, or explore distillation.

Need for Immediate Control

The Problem: Fine-tuning hardcodes behavior. You lose the ability to instantly "patch" a harmful or flawed response with a prompt change.



For high-stakes apps, keep logic in prompts or use external rule-based guardrails for maximum control.

@shivanivirdi

NEOSAGE

Part 2: The Execution Playbook

Once you've made the call, execution becomes everything.

This is where most teams fail, trying to brute-force results without a loop.

Your job is to build a Minimum Viable Model that fails in informative ways.

@shivanivirdi

NEOSAGE

The Fine-Tuning Loop

Successful fine-tuning is an iterative engineering discipline.

Define the Task

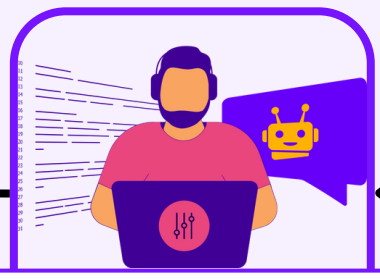
Get crystal clear on the specific behavior you are trying to teach.

Goal: Minimum Viable Model (MVM), not perfection.

Curate the Dataset

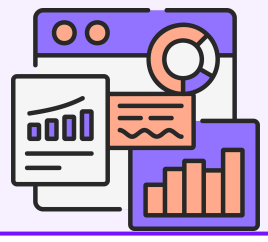
Build a small, high-quality dataset that perfectly represents the target behavior.

Choose Method & Train



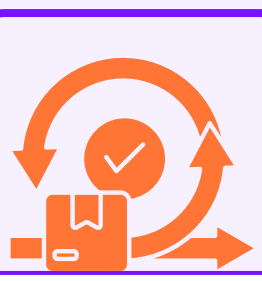
Select a technique (e.g., PEFT) and run the training job to create a baseline model.

Evaluate



Rigorously test the model's behavior to find where it succeeds and, more importantly, where it fails.

Refine & Repeat



Analyze failures, improve the dataset with new, targeted examples, and begin the loop again.

Every failure is a high-quality signal for the next iteration.

@shivanivirdi

Data is the Leverage

Most engineers over-index on methods.
But the real power lies in data quality.

- 1,000 clean examples > 50,000 noisy ones
- Your dataset is the source code of your model's behaviour
- Format, tone, and reasoning, all encoded in examples

The Anatomy of a Golden Example

The Instruction
(The Task)

```
You are a support agent who must politely decline feature requests that are not on the company's public roadmap.
```

A clear, unambiguous prompt that defines the goal.

The Context
(The Input)

```
User query: "Will you be adding support for interplanetary communication protocols next quarter?"
```

The specific information the model needs to act on.

The Completion
(The Target Behavior)

```
That's a fascinating question! While we're focused on terrestrial communication for now, I really appreciate your forward-thinking suggestion. I'll pass it along to the team for future consideration. Is there anything else I can help with
```

The ideal response, perfectly modeling the desired tone, format, and reasoning.

@shivanivirdi

NEOSAGE

Fine-Tuning Methods

A high-level guide to the trade-offs between the core training techniques.

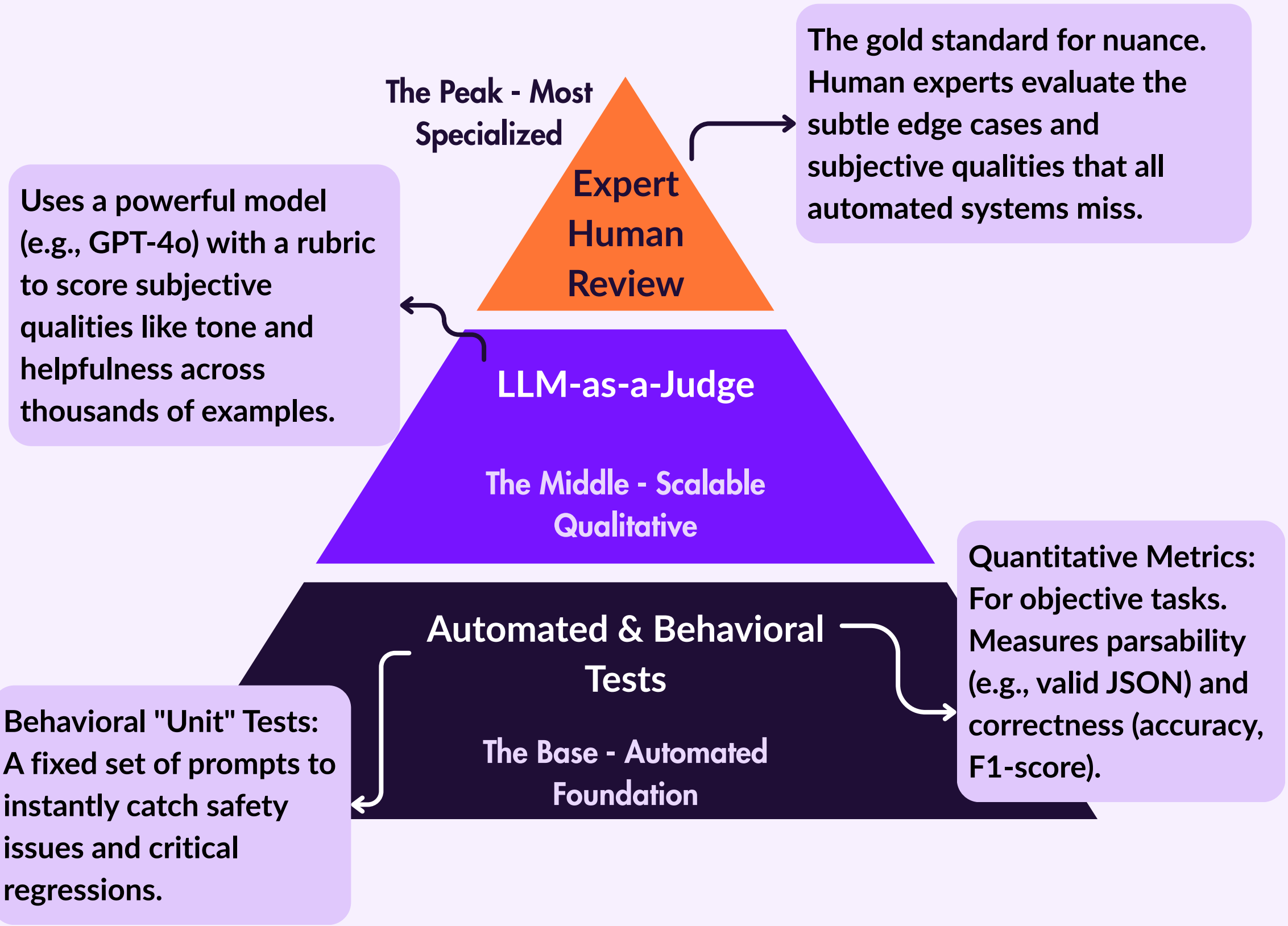
Criteria	Full Fine-Tuning (SFT)	Parameter-Efficient Tuning (PEFT)	Preference Tuning (DPO)
Training Cost	Requires significant, multi-GPU clusters and is prohibitively expensive for most.	Highly efficient; can often be trained on a single consumer GPU, making it widely accessible.	Compute cost is moderate, but the primary expense is in generating high-quality human preference data.
Dataset Size	Needs a very large, clean dataset (10k-100k+ examples) to avoid severe overfitting.	Flexible and effective with smaller, targeted datasets in the hundreds to thousands of examples.	Requires thousands of preference pairs (chosen vs. rejected) which are labor-intensive to create.
Flexibility / Power	Offers maximum power to learn new, complex skills that are distant from the base model's training.	Excellent for adapting a model's style, format, or domain knowledge while preserving its core reasoning.	Uniquely suited for aligning a model to subjective qualities like helpfulness, tone, and safety.
Risk Profile	High risk of catastrophic forgetting and overfitting if not managed with extreme care.	Low risk to the base model, as its core weights are frozen and protected from being overwritten.	Medium risk; does not guarantee factuality, as it optimizes for being preferred, not correct.

@shivanivirdi

NEOSAGE

A Modern Evaluation Stack

A modern evaluation stack uses a portfolio of tests to measure what truly matters: behavior.



A robust strategy uses every layer. They work together to catch different kinds of failures.

@shivanivirdi

NEOSAGE

Defend Before You Deploy

A practical checklist for mitigating common fine-tuning risks.

Failure Mode	Description	Prevention Strategy
Safety Alignment Collapse	Fine-tuning on benign data accidentally "drowns out" the model's original safety training, creating new vulnerabilities.	Use a behavioral "unit test" suite to monitor safety and refusals after every training run.
Catastrophic Forgetting	The model becomes an expert on your new task but loses its general knowledge and reasoning capabilities.	Use PEFT to protect base model weights; run regression evaluations against broad benchmarks (e.g., MMLU).
Overfitting & Mode Collapse	The model memorizes the style of your training data so perfectly that its responses become generic and repetitive.	Ensure dataset diversity with varied examples; tune for fewer epochs (often 1-3 is enough).
Bias Amplification	The model learns and exaggerates social or demographic biases present in your training data, leading to unfair outputs.	Audit your dataset for representation skews before training; use slice-based evaluations to test for inequitable performance.

@shivanivirdi

NEOSAGE

Final Note

Fine-tuning is not a boost button.
It's a **surgical override**, and with that power comes risk.

If you're serious about shipping reliable LLM products, this two-part guide will save you months of trial and error.

Part 1: The Strategy → https://blog.neosage.io/p/an-engineers-guide-to-fine-tuning

Part 2: The Execution Playbook → https://blog.neosage.io/p/an-engineers-guide-to-fine-tuning-cd5

@shivanivirdi

@shivanivirdi

Liked This?

SAVE
REPOST
FOLLOW

Read full issue on my newsletter blog.neosage.io

NEOSAGE