

Info for Last Minute Deciders - Overview of Fine Tuning Locally Run AI models



🔧 Fine Tuning LOCAL models

- Agentic, RAG, LoRA, RL
- Fine tuning techniques discussed here are only possible on local models*

🔧 What we will do

- Discuss 4 methods of fine tuning models
- Pros and cons
- Go through some code
- Github has working examples for each method

⚠️ Coding Agents not included

- They're designed to be agents already.

Overview of Fine Tuning Local LLM Techniques

Making sense of the different ways to improve LLMs and fix hallucinations

Michael Ramshaw
RAG section by Kai Ebira

Agents

LoRA

RAG

RL



Center for

HIGH PERFORMANCE COMPUTING

THE UNIVERSITY OF UTAH

hallucination
repair kit

Why tune at all?

LLMs are powerful, complicated, and occasionally tiny **goblin** orchestras.



Harmless weirdness (at first...)

- WSJ report – GPT 5.4 suffered a goblin invasion. Mentions of goblins increased almost 4000%.
- “Fixed” in March with new update. Wasn’t fixed, goblins won again
- Problem was still present in 5.5 last week
- THIS IS A COMPLEX PROBLEM!

Real risk

- Hallucinations
- Knowledge gaps (ESPECIALLY SITE/DOMAIN)
- Sycophancy
- Unsafe/illegal behavior

Googling “OpenAI lawsuits” will not come up empty.

Local control

With local models, you can (try) to fix these issues

Why local LLMs? It's definitely more work...

<https://mitsloan.mit.edu/ideas-made-to-matter/ai-open-models-have-benefits-so-why-aren't-they-more-widely-used>

Open model snapshot from MIT

Closed model usage **~80%**

Open model performance **~90% at release**

Closed model cost **~6x open**

NCAR pays for Claude, looking into switching to local



Reddit r/LocalLLM

- “If people understood how good local LLMs are getting”
- “Anyone else feel like LLMs aren't actually getting that much better?”
- Earlier talks today from BSU and MSU show adoption



Benefits

- Keep data local (HIPAA, security)
- Avoid vendor lock-in
- Many model choices
- lower running cost
- **Retrain or adapt when needed**



Drawbacks

- Not always bleeding edge
- Frozen at release
- Open weights \neq open datasets
- Fewer guardrails
- Customization effort
- Significant GPU requirements
- Constant changes

<https://www.wsj.com/tech/ai/chatgpt-became-so-obsessed-with-goblins-that-openai-had-to-intervene-a6644e66>

Four routes to a CHPC – University Chatbot

Each method has pluses and minuses

Agentic AI/MCP

Adds tools, skills,
MCP servers

Base model is
untouched

LoRA

Trains and changes
specific layers of
model, NOT all

RAG

Adds external
context at inference
time

Base model is
untouched

RL

Optimizes behavior
with reward signals

Significant training

The scoring lens

We did LoRA and RAG to add CHPC-specific information to our chatbot.



GOODNESS

- Does it give localized answers?
- Does it show hallucination?
- Does it give the best answer as good as a human would?



EFFORT

- Learning LLM programming
- Compute
- Data Wrangling
- Human labeling are required



DIFFICULTY

- Implementation
- Maintain
- Keep compatible as libraries change

Agentic AI / MCP vs. AI agents

The line is fuzzy...

Agentic AI

- Supplementary material for a model
- NO ACTIONS TAKEN
- Update model with constantly updated knowledge
- Many MCP servers available or make your own
- Can customize further with skills

AI Agents – what we're not talking about

- Agents taking actions on your behalf
- ACCESS A BIG ISSUE
- Coding Agents – we allow these in GE
- Openclaw – we do not allow this
- Booking travel, choosing flights

Agentic AI / MCP: What we are talking about

Task was to make a chatbot specific to CHPC – Agentic can do many more things and we didn't even mention Skills files

Agentic AI

- Supplements knowledge of base model
- Up to the second info (e.g. the time or your favorite team results)
- No model training but adds MCP access time like RAG
- For this comparison, the agent was a DuckDuckGo web search library.
- Easy – no data wrangling

Model Context Protocol (not Master Control Program)

- Open standard interface for external tools
- Lots of MCP servers out there
- Can be clients or servers
- Different provider libraries exist
- Do you want to allow internet access?

Skills

- Task-specific instructions and know-how
- Good for shaping reasoning without wiring an operational tool

https://github.com/chpc-uofu/AIML_Intro/tree/main/LLM_Fine_Tuning_Presentation

Worst – base model by itself with no help

Starting with base knowledge, there is no CHPC Colorado...

What GPUs are available at CHPC, and how can I access them?

OI `/scratch/general/vast/app-repo/huggingface/Qwen/Qwen3-8B`

Thought for 1 seconds ▾

I don't have access to real-time hardware inventory or access policies for external organizations like CHPC (Colorado High Performance Center). For the most accurate information about available GPUs and access procedures, I recommend checking:

1. CHPC's official website
2. Their documentation portal
3. Contacting their support team directly

Would you like me to help you find any documentation or resources about CHPC's GPU resources?



Prompts alone are not enough

If there are questions about CHPC at the University of Utah, search their documentation to find the answer.

OI `/scratch/general/vast/app-repo/huggingface/Qwen/Qwen3-8B`

Thought for 1 seconds ▾

View Result from `query_knowledge_files` ▾

Thought for 1 seconds ▾

View Result from `search_knowledge_bases` ▾

View Result from `search_knowledge_files` ▾

Thought for 1 seconds ▾

It seems there are no available documentation resources for CHPC at the University of Utah in the current system. To proceed, you might want to:

1. Check the University of Utah's official CHPC website directly
2. Contact their support team for documentation access
3. Verify if the documentation exists in a specific knowledge base you have access to

Would you like me to help with any of these alternatives?



Better – MCP server added

Added MCP server to add web search capability. We don't allow Docker and CHPC link is for South Africa

What GPUs are available at CHPC, and how can I access them?

OI /scratch/general/vast/app-repo/huggingface/Qwen/Qwen3-8B

Thought for 1 seconds ▾

View Result from DDGS_duckduckgo_search ▾

Thought for 2 seconds ▾

To determine the GPUs available at CHPC and how to access them, follow these steps:

1. Check CHPC Modules:

Use the command `module avail chpc` to list available modules. Look for GPU-related subgroups (e.g., `chpc/earth` for earth-science applications) or specific GPU drivers. This will help identify supported GPU software stacks.

2. GPU Access via Docker:

If using Docker, allocate GPUs with the `--gpus` flag (e.g., `docker run --gpus all ...`). However, ensure this is supported by CHPC's Docker environment.

3. Consult CHPC Documentation:

Visit [CHPC's Survival Guide](#) for system-specific details. If GPU models or access procedures are unclear, contact CHPC support for guidance.

4. System-Specific Resources:

For systems like Northwestern's Quest, GPU access often requires requesting quotas or using interactive sessions. Verify if CHPC has similar processes.

For precise details on available GPUs and access methods, reaching out to CHPC's support team or checking their official documentation is recommended.



A little help - Prompt it to search for CHPC and Utah

Prompt links CHPC to University of Utah, it does much better now with web search agent.

If there are questions about CHPC at the University of Utah, search their documentation to find the answer.

O1 /scratch/general/vast/app-repo/huggingface/Qwen/Qwen3-8B

Thought for 2 seconds ▾

View Result from DDGS_duckduckgo_search ▾

Thought for 1 seconds ▾

To find answers about the **University of Utah's CHPC (Center for High Performance Computing)**, refer to the following key resources from their official documentation:

1. Official CHPC Documentation

URL: <https://www.chpc.utah.edu/documentation/>

Content:

- **Getting Started Guide:** Instructions for accessing CHPC resources.
- **Resource Access:** Details on GPU usage, storage, and software modules.
- **Support:** Contact information for assistance.

2. Storage Options at CHPC

URL: [Storage-CHPC.pdf](#)

Content:

- Information about file storage options, including group space for sponsored research projects.



3. QES Software Installation (Optional)

Best – The Answer We Want

app-repo/nuggingrace/Qwen/Qwen3-8B ▾ ↑

What GPUs are available at CHPC, and how can I access them?

OI /scratch/general/vast/app-repo/huggingface/Qwen/Qwen3-8B

Thought for 1 seconds ▾

View Result from DDGS_duckduckgo_search ▾

Thought for 1 seconds ▾

To find out what GPUs are available at the **University of Utah's CHPC (Center for High Performance Computing)** and how to access them, refer to the official documentation:

1. Official GPU Resources at CHPC

URL: [GPUs and Accelerators at CHPC](#)

Key Details:

- **Available Clusters:** GPU-enabled systems include `granite`, `notchpeak`, `kingspeak`, `lonepeak`, and `redwood` (part of the Protected Environment).
- **GPU Models:** The document specifies hardware details (e.g., NVIDIA Tesla/V100, A100, or other models).
- **Access Procedures:**
 - Use `module avail` to check GPU-specific software stacks.
 - Submit SBATCH jobs with GPU allocation (e.g., `SBATCH --gres=gpu:4` for 4 GPUs).
 - Contact CHPC support for advanced configurations or quotas.

2. How to Access GPUs

- **Step 1:** Load GPU modules using `module load cuda/11.8` (or the version available).
- **Step 2:** Submit SBATCH jobs with GPU resources:

```
bash
```

```
1 sbatch --gres=gpu:4 my_job.sh
```

⌵ Collapse Copy

- **Step 3:** Use GPU-aware tools like `nvidia-smi` to monitor usage.

Agentic evaluation: the good and the bad

Prompting is important (a theme for all of these)

✓ The good

- Localized most of the time
- Functional answers ~60%
- Easiest to set up
- No data wrangling OR model retraining
- Model freedom and interchangeability
- ChatGPT will let you use MCP servers now (Costs more)

⚠ The bad

- Worst out of all methods*
- Most hallucinations
- Rarely got best answer
- Additional inference time to contact MCP server

*** MCP servers can be written by you or use what's out there. Can always try again.**

LoRA (Low Rank Adaptation)

Low-Rank Adaptation modifies a base model with small domain adapters.

Use case

Add organization-specific or updated **domain** knowledge to a local model without full retraining.

What you need

- Base model from Hugging Face (we have many)
- a dataset, for example a two-column prompt/completion CSV (translated from Vector Store)

Pluses

- Domain-specific training
- fast inference
- adapters per domain
- update as needed
- preserves base knowledge
- Small storage space (e.g. 400M for 55G)

Minuses

- Retraining time
- Frozen again after training
- Learning curve
- Too many hyperparameters
- Models add new trainable layers
- support can lag new model architectures

https://github.com/chpc-uofu/AIML_Intro/tree/main/LLM_Fine_Tuning_Presentation

LoRA evaluation: the good and the bad

Early results: promising, but phrasing-sensitive.

The good

- Localized in all cases without prompting but did even better with.
- Faster answers – no external retrieval – new info baked in model via adapter
- Functional answers around 2/3 of the time
- Very good answers in about one-third of cases
- Better than expected for a first try – there are many modifications that could be done

The bad

- Phrasing mattered, “At CHPC” improved answers dramatically
- Programming had a learning curve but not near as bad as RL
- LoRA support can lag new model architectures in libraries.
- Lora Adapters NOT transferable between models

RAG: Retrieval Augmented Generation

Retrieval-Augmented Generation feeds the model relevant external context.

The Process

- Scrape.py - Scrape our website documentation
- Ingest.py – Huggingace embeddings into a Qdrant Vector Store database
- Query.py - retriever searches a knowledge base
- Start.sh – start it all

Pluses

- No model retraining – model is untouched
- Knowledge can be updated by re-ingesting the vector store
- Localizes answers and can reduce hallucination/generic answers
- Vector store can be used by any open source model

Minuses

- Setup and data wrangling – Data quality > model size
- Higher runtime resources
- Slower inference because retrieval happens first.
- A LOT of work
- Results comparable to other methods even with strict prompts saying to only use vector store

<https://github.com/kaiebira/CHPC-ollamaRAG/tree/main>

RAG evaluation: the good and the bad

Early results: promising, but phrasing-sensitive.

The good

- Localized when very specific prompting was used.
- Functional answers around 2/3 of the time
- Very good answers in about one-third of cases
- Vector Store can easily be used with different models

The bad

- Wrong pathnames for modules and examples
- Most complicated to do
- Re-ingesting can be complicated (chunking)

Reinforcement Learning

Encourage the behavior you want and discourage the behavior you do not.



Use case

- Change undesired behavior of a model
- Perhaps your model was trained with bad data
- Maybe you don't know why it's acting like this



Workflow

- Base model
- Set of prompts that reveal bad behavior
- Reward function/model/rule → promote good, discourage bad
- Training loop like LORA but dataset needs scores or some kind of eval

LORA Similarities / Differences

- Training a model
- Value is the key difference
- Entire model must be saved
- Framework is evolving

Reinforcement Learning

Encourage the behavior you want and discourage the behavior you do not.



Pros

- Ideally your model no longer has undesired behavior
- No more hallucinations
- No more ethical issues
- No more toxicity
- Goblin invasion is repelled



Cons

- Lots of memory on GPU required
- Lots of storage required (saves full model)
- Libraries change way too fast
- Training can destroy model knowledge
- Difficult to measure success
- Difficult to do at all
- Even OpenAI can't get this right













Conclusion

- Leave this to model makers if possible
- Easy to break the whole model
- Libraries move under your feet

https://github.com/chpc-uofu/AIML_Intro/tree/main/LLM_Fine_Tuning_Presentation

Scorecard: what should you try first?

Common theme – All require some prompting except RL

Method	Best for	Goodness	Effort	Difficulty
LoRA	Domain adaptation, localization			
RAG	Fresh facts / documentation			
Agentic/MCP	Very fresh facts, Tools, APIs, workflows			
RL	Behavior shaping			

TL;DR

Try Agentic first. It is easiest to setup but worst (for us anyway). RAG and LoRA are better but require more effort. Avoid RL unless behavior alignment is the core job.

But what does “goodness” really mean?

<https://huggingface.co/spaces?sort=trending&search=leaderboard>



Metrics

- There are many (reasoning, math, etc)
- Goodheart’s Law has been brought up a lot



Homemade is best

- Task-specific
- Make a rubric or set of evaluation questions

Your metric is most important!



Operational cost

Latency

GPU memory

Storage

Maintenance

Labeling

Evaluation effort

Grand finale (Remember prompting is critical!)

Current info?

Agent/MCP

- Some prompting needed
- No model retraining!
- Adding web search can solve a lot of issues.
- Works with ChatGPT
- Easiest to setup

Missing facts?

RAG

- Some prompting needed
- Great for documentation cases
- No model retraining!
- Work will be required
- Data Wrangling

Domain knowledge?

LoRA

- Less prompting needed
- Partial model
- Training required
- Data wrangling
- Hyperparameters
- Small storage

Bad behavior?

RL

- Difficult to do and difficult to measure
- Full retraining required
- Large storage requirements
- Data Wrangling (even worse)

You can combine these or do whatever you want.

Questions?



Mike's Github (all but RAG)
michael.ramshaw@utah.edu

Kai's Github (RAG)
Kai.Ebira@utah.edu

