Coding Nexus · Follow publication

Member-only story

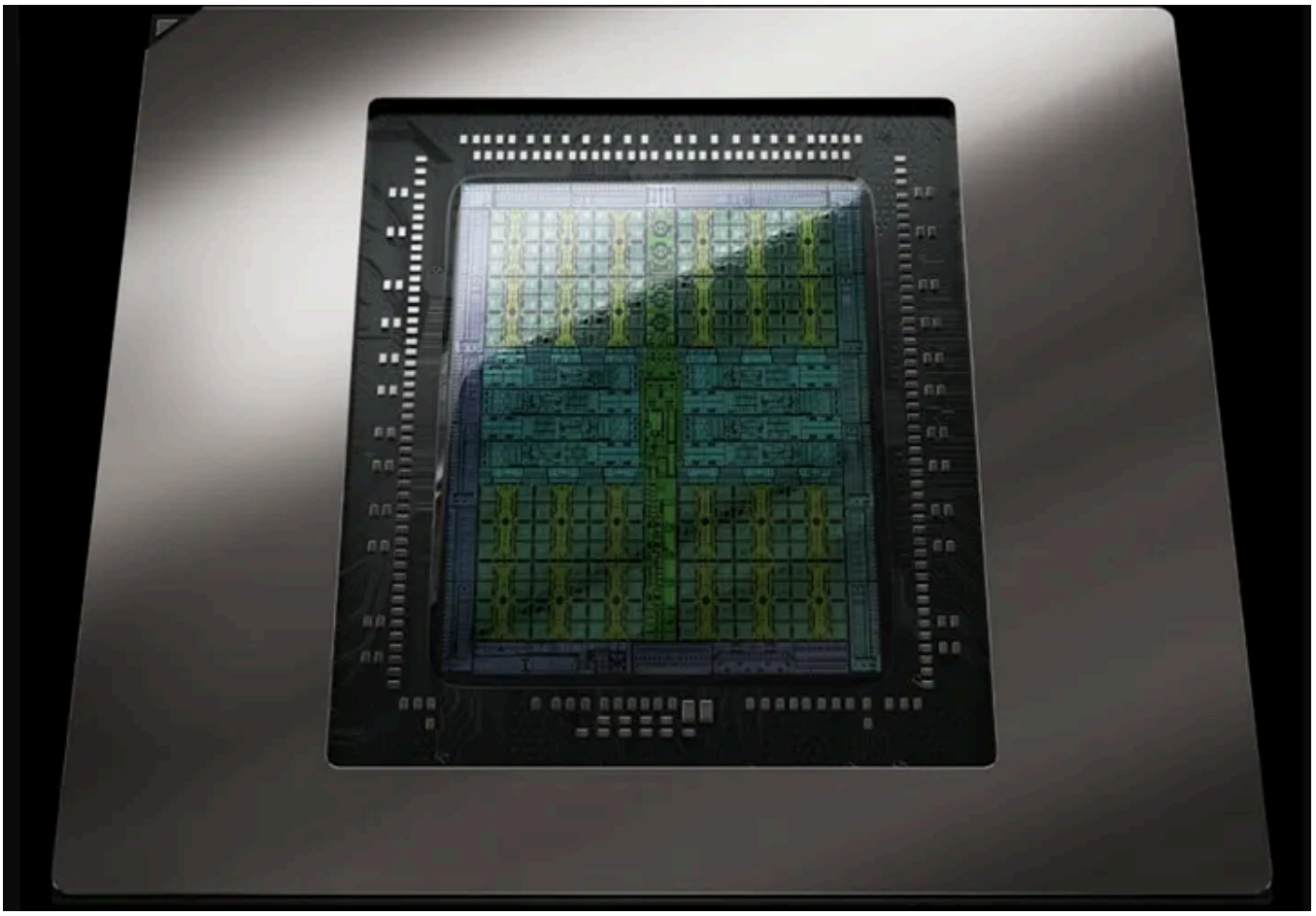
I Trained an LLM on My RTX 5090 with Unsloth

4 min read · 1 day ago

Civil Learning Following

Last weekend, I achieved something I didn't think was possible a year ago — I fine-tuned a 20B language model on my own desktop GPU.
No cloud credits. No cluster. Just me, an RTX 5090, and this open-source project called **Unsloth**.

I've been following NVIDIA's **Blackwell** architecture launch for a while, and everything about it screamed "AI playground." So when I saw that Unsloth added Blackwell support, I thought: *let's see how far a single GPU can go.*



RTX 5090

. . .

What the hell is Unsloth?

If you've ever tried fine-tuning a large model, you know it's usually a nightmare — hours of configuration hell, VRAM errors, and "CUDA out of memory" messages that make you want to throw your PC out the window.

Unsloth addresses many of those issues. It's an open-source framework that essentially optimizes the core processes of fine-tuning and reinforcement learning for LLMs. Think: **2× faster training**, **70% less VRAM**, and **no accuracy loss**.

It supports popular models like Llama, gpt-oss, and DeepSeek. And the coolest part? It's now optimized for **NVIDIA Blackwell GPUs** — including the RTX 50 series, the new RTX PRO 6000, and DGX Spark systems.

So yeah, you can actually start fine-tuning 20B or even 40B models at home.

. . .

The setup

I expected dependency chaos. Instead, I ran:

```
pip install unsloth
```

...and it just worked.

Then I switched to Python:

```
from unsloth import FastLanguageModel
import torch

model, tokenizer = FastLanguageModel.from_pretrained(
    model_name = "unsloth/gpt-oss-20b",
    max_seq_length = 1024,
    load_in_4bit = True,
    full_finetuning = False,
)

from unsloth import FastLanguageModel
import torch
max_seq_length = 1024

fourbit_models = [
    "unsloth/gpt-oss-20b-unsloth-bnb-4bit",
    "unsloth/gpt-oss-120b-unsloth-bnb-4bit",
    "unsloth/gpt-oss-20b",
    "unsloth/gpt-oss-120b",
]

model, tokenizer = FastLanguageModel.from_pretrained(
    model_name = "unsloth/gpt-oss-20b",
    max_seq_length = max_seq_length,
    load_in_4bit = True,  # keeps VRAM low
    full_finetuning = False,
   )
```

Boom — model loaded successfully, with no CUDA errors.
That alone felt like a victory.

. . .

Blackwell + Unsloth

Here's where it became interesting.

With my RTX 5090 (32GB VRAM), fine-tuning a **20B model** proved *actually feasible*. The training speed was about **twice as fast as the** setup I used with Hugging Face and Flash Attention 2. VRAM usage decreased by roughly **70%**, and I was able to increase the **context window to 122k tokens**.

That's not a typo — 122,181 tokens on a single consumer GPU. Before Unsloth, I'd hit "out of memory" with anything over 4k.

Here's what the numbers looked like:

VRAM	Unsloth Context Length	HF + FA2 Context
8 GB	2,972	OOM
16 GB	40,724	2,551
32 GB	122,181	9,711

You can actually run Llama 3.1 8B on a mid-range GPU and still have extra capacity.

. . .

Scaling up — without changing code

The magic here isn't just about local training.
Unsloth was designed to scale from your desktop GPU to **NVIDIA DGX Cloud** or any **NVIDIA Cloud Partner** setup.

Same code, same config — you switch the hardware.
So when you're done experimenting locally, you can train 70B+ models on Blackwell clusters without touching a single line of Python.

. . .

If you hit xFormers issues...

You might encounter build errors with xFormers (I did). The solution is straightforward:

```
pip uninstall xformers -y
pip install ninja
export TORCH_CUDA_ARCH_LIST="12.0"
git clone --depth=1 https://github.com/facebookresearch/xformers --recursive
cd xformers && python setup.py install && cd ..
```

After that, smooth sailing.

. . .

Docker option

If you prefer containers (which I do for production), Unsloth offers a ready-to-use Docker image.

```
docker run -d -e JUPYTER_PASSWORD="mypassword" \
  -p 8888:8888 -p 2222:22 \
  -v $(pwd)/work:/workspace/work \
  --gpus all \
  unsloth/unsloth
```

You'll need to install the **NVIDIA Container Toolkit**, but once it's set up, you'll have a full Jupyter lab ready to fine-tune large models.

```
python -m venv unsloth
source unsloth/bin/activate
pip install unsloth
```

. . .

Why this matters

There's a quote from Unsloth co-founder **Daniel Han** that stuck with me:

> "AI shouldn't be an exclusive club. The next great AI breakthrough could come from anywhere — students, individual researchers, or small startups."

That's precisely what this setup signifies — the democratization of LLM fine-tuning. We're now at a stage where a kid with a gaming PC can train a model that once needed a data center. That's wild.

. . .

Final thoughts

If you've got a **RTX 5090** or **RTX PRO 6000 Blackwell**, you can start small today.
Then, when you're prepared to scale, move to **DGX Cloud** — no refactoring, no fuss.

I'll say it: Unsloth makes fine-tuning *enjoy*able once more.
And with Blackwell GPUs, it feels like we've entered a new era — where "training an LLM" no longer means "renting an AWS cluster."

. . .

TL;DR:
Unsloth + NVIDIA Blackwell = local fine-tuning for everyone.
2× faster training
70% less VRAM
12× longer context
No cloud dependency

Unsloth | Llm | Nvidia | Llm Applications | Llm Agent

Follow

Published in Coding Nexus

8K followers · Last published 8 hours ago

Coding Nexus is a community of developers, tech enthusiasts, and aspiring coders. Whether you're exploring the depths of Python, diving into data science, mastering web development, or staying updated on the latest trends in AI, Coding Nexus has something for you.

Following

Written by Civil Learning

3.1K followers · 6 following

We share what you need to know. Shared only for information.

No responses yet

Bgerby

What are your thoughts?

More from Civil Learning and Coding Nexus

In Coding Nexus by Civil Learning

MarkItDown: Convert Anything into Markdown — the Smart Way to Feed LLMs

You know that feeling when you're trying to feed a PDF or a Word document into an LLM, and it just doesn't understand what's going on...

Oct 15 245 4

In Coding Nexus by Civil Learning

The Guy Who Let ChatGPT Trade for Him — and Somehow It Worked

You know how everyone says, “Don’t let AI touch your Money”? Well, someone on Reddit decided to ignore that.

Oct 8 178 8

In Coding Nexus by Algo Insights

4 Open-Source Tools That Made Me Rethink My Dev Setup

I’ve been coding for a while now. Most of the tools we use every day… they’ve been the same for years. Editors, browsers, frameworks. Just…

Sep 3 451 9

In Coding Nexus by Civil Learning

Google’s New LLM Runs on Just 0.5 GB RAM—Here’s How to Fine-Tune It Locally”

A few days ago, Google quietly released a little AI model called Gemma 3 270M.

Aug 15 2.1K 30

See all from Civil Learning

See all from Coding Nexus

Recommended from Medium

In Towards AI by Gao Dalie (高達烈)

RAG is Not Dead! No Chunking, No Vectors, Just Vectorless to Get the Higher Accuracy

Over the past two years, I have written numerous articles on how Retrieval-Augmented Generation has become a standard feature in nearly all...

Oct 17 730 6

In Coding Nexus by Algo Insights

Local LLMs 101: What Really Happens When You Run an AI Model on Your Own Machine

If you've ever tried running a large language model locally—and your GPU fans started screaming—you've probably wondered: what's...

3d ago 133 1

Doran Gao

Getting Started with NVIDIA DGX Spark: Open WebUI & ComfyUI Setup Guide

“To innovate is to disrupt.” — Theodore Levitt.

Oct 18 30 1

In Towards Deep Learning by Sumit Pandey

Why Everyone Will Want DGX Spark on Their Desk — Yes, Everyone

I just saw this picture today and was amazed, I’ve been waiting for this moment for a long time. (No, it’s not Elon.) It’s that tiny...

Oct 14 70 11

In AI Software Engineer by Joe Njenga

This Viral DeepSeek OCR Model Is Changing How LLMs Work

This DeepSeek OCR model hit an overnight success not seen in any other release—4k+ GitHub stars in less than 24 hours and more than 100k...

Oct 23 287 4

Ignacio de Gregorio

LLMs are smarter than we thought.

Changing how they talk changes how they think.

2d ago 729 22

See more recommendations