

Vraj Patel

774-627-0286 | vrajpatel.jobs@gmail.com | [LinkedIn](#) | [GitHub](#) | [Portfolio](#) | [Blog](#)

EDUCATION

University of Massachusetts Dartmouth / North Dartmouth, MA

Expected May 2027

Bachelor of Science in Computer Science | GPA: 3.53

Relevant Coursework: Data Structures & Algorithms, Probability, Machine Learning, Theory of Computation, Software Engineering

TECHNICAL SKILLS

Languages: Python, C++, CUDA, SQL

Inference Systems: Paged KV Cache, Continuous Batching, FlashAttention-2, INT8 Quantization (KV cache, W8A8, W8A16), ONNX

GPU & Performance: PyTorch, Triton, CUDA, Nsight Compute/Systems, latency and throughput benchmarking (TTFT, TPOT, tok/s)

Model Architectures: Transformers (RoPE, GQA, MoE, RMSNorm, SwiGLU), LSTM/GRU, RL (SAC)

Serving & Infrastructure: FastAPI, Docker, GitHub Actions CI/CD, AWS (EC2, S3, ECR, ECS), Snowflake, Linux, Git

AI Coding Tools: Claude Code (daily use; led a cohort workshop on AI-assisted development), Gemini (Antigravity)

PROJECTS

paged-inference-engine - vLLM-Style LLM Inference Engine From Scratch / *PyTorch, Triton*

[GitHub](#)

- Eliminated KV-cache fragmentation across concurrent requests by implementing **PagedAttention** from the paper without referencing vLLM source: a block allocator with per-sequence block tables over one fixed GPU memory pool.
- Kept the GPU busy across 1 to 32 concurrent requests by building a **continuous-batching scheduler** that admits new requests mid-batch as KV blocks free up, with lookahead admission and a skip-threshold fallback; unit-tested under memory pressure.
- Matched PyTorch SDPA **within 0.001** at FP16 on scrambled, non-contiguous block layouts by rewriting a **FlashAttention-2 Triton kernel** to gather K/V through a block table with in-kernel causal masking.
- **Halved KV-cache memory** with output identical to the FP16 path by adding per-block, per-head **INT8 KV quantization** with dequantization fused into the kernel attention loop.
- Raised throughput **1.85x** (95 to 176 tokens/s, 1 to 32 concurrent requests) in a CUDA-synchronized TTFT/TPOT harness; profiled the gap to a sequential baseline to per-step launch, block-table, and padding overhead.

paged-engine-serving - Serving Layer for the Inference Engine / *FastAPI, Docker, GitHub Actions, AWS*

[GitHub](#)

- Served concurrent HTTP requests in the same GPU forward pass by wrapping the engine in **FastAPI** (/generate, /health, /metrics) with the engine loop on a background thread, exposing free KV blocks and queue depth.
- Cut the Docker image **33% (26 GB to 17.4 GB)** and kept cold containers out of rotation by gating /health on Triton JIT warmup, loading checkpoints from S3, and shipping SHA-tagged images to AWS ECR through GitHub Actions CI/CD.

edge-quantization-snapdragon - Model Bring-Up on a Snapdragon NPU / *PyTorch, ONNX, Qualcomm AI Hub*

[GitHub](#)

- Matched PyTorch on **99.5% of top-1 tokens** on device by bringing up a from-scratch transformer on a **Snapdragon 8 Elite Gen 5** NPU (PyTorch to ONNX to QNN) with every layer running on the NPU.
- Cut on-device latency **44% (2.5 to 1.4 ms)** and peak memory **about 3x** with W8A8 quantization, and isolated its accuracy loss to LayerNorm outputs and the residual stream, not the Linear layers, by running a per-tensor sensitivity sweep.

cpg-gpu-inference - GPU Kernels and Quantization From Scratch / *CUDA, Triton, PyTorch, C++*

[GitHub](#)

- Held peak memory level with PyTorch SDPA from 512 to 8K tokens by writing **FlashAttention-2 forward and backward** in Triton as a custom autograd Function that never materializes the full attention matrix.
- Cut static model memory **46.4%** (vs. 29.3% for bitsandbytes) with perplexity **within 0.02%** and 15 of 15 outputs identical by reimplementing LLM.int8() outlier-aware quantization from scratch.
- Reached **within 20% of cuBLAS** (512 to 4096, true FP32) with an autotuned Triton matmul, and cut uncoalesced-load loss from **43% to 3%** (Nsight Compute) by tuning CUDA kernels for coalescing and shared-memory tiling.

NutriGrove - LLM-Powered Mobile App / *Flutter, FastAPI, Firebase, Gemini API*

[GitHub](#)

- Reached **100+ users on the App Store and Google Play** by shipping a full-stack app whose FastAPI backend calls Gemini to plan meals from live dining-hall menus; **2nd place** at a \$12K AI hackathon.

EXPERIENCE

Fleet Robotics - ML Engineer Intern / *Boston, MA*

Jun 2026 - Aug 2026

- Found why a Soft Actor-Critic (SAC) navigation policy failed on the real robot (simulated drift near 0 degrees vs. about 20 on hardware) by tracing it to a MuJoCo environment that did not accumulate noise across steps.
- Cut the sim-to-real gap **from about 20 degrees to about 3** by rebuilding the noise model against AprilTag ground truth from 20 real robot runs and fine-tuning SAC on the corrected simulator.
- Turned raw robot telemetry into engineering insights in a **roughly 500K-line codebase** by building AWS/Snowflake pipelines and Streamlit dashboards; presented results to leadership; used Claude Code to speed up iteration.

General Dynamics Mission Systems - Senior Capstone Team Member / *UMass Dartmouth*

Sep 2026 - Present

- Training LSTM, GRU, and transformer state estimators and benchmarking them against an **Extended Kalman Filter (EKF)**.

SMAST, UMass Dartmouth - Research Software Engineering Intern; Data Science Intern / *Dartmouth, MA*

Jun 2025 - Nov 2025

- **Doubled SSEE simulation speed** via parallel processing; cut report turnaround from **3 weeks to 10 minutes** with ETL pipelines.