

PRATHAMESH HEDAU

Inference Engineer

prathamesh0404@gmail.com | +91-9702581533 | [LinkedIn](#) | [GitHub](#) | [Portfolio](#) | [X / Twitter](#)

SUMMARY

I'm an Inference Engineer focused on making LLMs practical, efficient, and cost-effective under real hardware constraints. The goal isn't just to build a working system, but to optimize the complete execution pipeline—from core text-generation baselines and memory caching, to dynamic batching, token streaming, speculative speed boosts, and model compression.

SKILLS

Programming: Python, C++, C

Machine Learning & Deep Learning: PyTorch, TensorFlow, Scikit-learn, Transfer Learning, Model Evaluation, Cross-Validation

Inference & Serving: Ollama, FastAPI, vLLM, TGI, TorchServe, ONNX Runtime

Optimization: Quantization (GGUF, INT8, FP16), KV cache, dynamic batching, speculative decoding, model compression, semantic caching

Hardware awareness: CPU inference, memory hierarchies

Systems: Profiling (latency, memory, throughput), Docker, Git, Linux

PROJECTS

Kshipra — CPU-Only LLM Inference Engine

[GitHub](#) · [Blog](#)

- Built baseline autoregressive generation and KV-caching from scratch in PyTorch for GPT-2 on CPU, within an 8GB RAM budget.
- Benchmarked both stages over 5 runs: KV-caching raised throughput $5.75 \rightarrow 17.0$ tok/s ($\sim 3\times$) and kept per-token latency flat ($\sim 55\text{ms}$) instead of climbing past 200ms as sequence length grew — at a measured cost of $\sim 60\%$ higher peak RAM (800MB $\rightarrow$ 1.28GB).
- Roadmap ahead: batching, dynamic batching, streaming generation, early-exit decoding, speculative decoding (distilGPT-2 draft / GPT-2-medium target), and INT8 quantization.

Vahini — Semantic Caching for Local LLM Inference

[GitHub](#) · [Blog](#)

- Built a local semantic-caching layer for LLM inference using FastAPI, Ollama, Qwen, and vector similarity search to avoid redundant model calls.
- Cut repeated-query latency by up to $275\times$ versus re-running inference, and surfaced failure cases where responses above 94% embedding similarity were still incorrect — informing threshold tuning for reliability.

Traffic Violation Proctoring System — Helmet & Triple-Riding Detection

[Research Paper](#)

- Built a computer vision system to detect helmet use and triple riding on two-wheelers, improving traffic violation monitoring efficiency by roughly 92%.
- Published findings in IJIRSET (International Journal of Innovative Research in Science, Engineering and Technology), April 2025.

EDUCATION

Bachelor of Engineering (B.E.), Electronics and Telecommunication — Mumbai University

09/2021 – 06/2025

CGPA: 7.13 / 10

CERTIFICATIONS

IBM AI Engineering Professional Certificate

Coursera / IBM, 2025

- Capstone: Satellite Image Classification — CNN/ViT hybrid (PyTorch, Keras), 95%+ accuracy ([GitHub](#))
- Capstone: QA Bot Web App — RAG pipeline (LangChain, WatsonxEmbeddings, Chroma, Gradio) ([GitHub](#))