SAI TEJA SRIVILLIBHUTTURU

saiteja.srivillibhutturu@gmail.com | in/saitejasrivillibhutturu | github.com/saitejasrivilli | saitejasrivilli.github.io

SUMMARY

Senior Software Engineer with 4+ years building scalable, high-performance GPU-accelerated inference systems. Achieved 2.7× throughput and 2.8× latency improvements on multi-GPU clusters using PyTorch, TensorRT, and ONNX Runtime. Led distributed systems handling 1M+ requests with 99% uptime. Skilled in optimizing prediction throughput under latency constraints using FlashAttention-2, quantization, and KV-cache techniques. Experienced in developing robust, open-source software for distributed AI inference workloads..

EDUCATION

Master of Science in Computer Science | University of Texas at Arlington **08/2023 - 05/2025**
Courses: Machine Learning, Neural Networks, Artificial Intelligence, Scalable Search and Optimization **GPA: 4.0/4.0**
Bachelor of Technology in Computer Science | Andhra University **06/2015 - 04/2019**

SKILLS

Languages & ML Frameworks : Python, C++, CUDA, PyTorch, TensorRT, ONNX Runtime, vLLM, FlashAttention-2, xFormers
Large Language Models: LoRA, Quantization (FP16, INT8, INT4-NF4, GPTQ, AWQ), Speculative Decoding, KV-Cache Optimization
Profiling & Debugging: torch.profiler, GPU Profiling, Memory Bandwidth Analysis, Roofline Analysis
Testing: Unit Testing, Integration Testing, Load Testing, A/B Testing, Regression Testing

EXPERIENCE

AI Inference Engineer | University of Texas at Arlington, TX **08/2024 - Present**

- Improved **GPU SM utilization** from 65% to 82% by debugging and profiling 12+ **inference configurations**, reducing **inference latency by 15%** using Python on 4-GPU A100 cluster for **CTMap** real-time navigation project
- Raised 2.1× training speedup and 63% memory reduction by fine-tuning **LLMs** using **PyTorch** FP16, LoRA, and gradient checkpointing
- Increased **inference throughput** by 2.7× (1.2K to 3.2K tokens/sec) by deploying vLLM with FlashAttention-2, KV-cache optimization, and **TensorRT** backend
- Built reusable **GPU profiling pipeline** with SLURM automation and **automated regression testing**, cutting benchmarking time by 80% across 200+ configurations/week

Machine Learning Engineer Intern | **ReplyQuickAI LLC**, **Remote, US** **12/2025 - Present**

- Fine-tuned YOLOv8s achieving 0.72 mAP@50 on multi-class dental detection across 8K+ images using **Python** and **PyTorch** on V100 GPU
- Deployed production model on AWS with Docker, exposing **HTTP REST** API with **JSON** responses, serving 10K+ daily requests at P95 latency 25ms and 99% uptime
- Designed **validation pipeline** with k-fold cross-validation and **A/B testing**, reducing false positive rate by 18% across 15+ deployments

System Engineer | Tata Consultancy Services, Chennai, India **06/2019 - 05/2023**

- Built and maintained RESTful APIs and gRPC services in Python and C++, adding input validation, retry mechanisms, and structured error handling, which lowered 5xx error rates by 30% for services handling 50K requests per day
- Diagnosed performance issues in distributed systems using profiling and debugging tools; mentored two junior engineers while identifying bottlenecks that improved API latency by 12% and reduced memory usage by 15%
- Coordinated high-scale microservices processing 10K+ events per hour across 10+ nodes, leading load and integration testing efforts that increased early bug detection by 25%
- Contributed to feature delivery across eight agile release cycles, performing code reviews with a focus on test coverage, performance benchmarks, and maintainability for a team of five engineers

PROJECTS

GPU Optimization & Profiling System | PyTorch, INT4-NF4, Mistral-7B gpu-optimization-mistral

- **Accelerated** 2.8× inference speedup and 4× memory reduction (14.4 to 3.6GB) by building integrated GPU optimization system combining kernel profiling, **C++/CUDA** optimization, and INT4 quantization with 0.1% accuracy loss
- **Improved GPU utilization from 68% to 91%** by profiling memory-bandwidth bottlenecks via roofline analysis and implementing fused kernels with gradient checkpointing, enabling 2.1× batch scaling for production deployment

Quantization & Speculative Decoding Benchmark quantization-speculative-decoding-benchmark

- Benchmarked 5 quantization methods (FP16, INT8, INT4-NF4, GPTQ, AWQ) with ONNX Runtime and TensorRT on LLMs; achieved **75% memory reduction, 3.3x inference speedup** while maintaining 99%+ accuracy
- Implemented speculative decoding with GPU profiler capturing kernel-level metrics and memory bandwidth; delivered **2-3x throughput improvement** with <0.2% accuracy loss

Attention Mechanism Optimization |FlashAttention-2, torch.profiler, CUDA attention-optimization

- Benchmarked 4 attention implementations (Vanilla, SDPA, FlashAttention-2, xFormers) on NVIDIA L4; achieved **12.3x throughput and 99.7% memory reduction** via IO-aware memory patterns
- Built batch size auto-tuner optimizing throughput under latency constraints; proved **algorithm-level optimization outperforms TensorRT by 6x** for LLM inference serving