Subhadeep Chatterjee

858-346-3765 | suchatterjee@ucsd.edu | linkedin.com/in/subhadeep19 | github.com/suga-ucsd | suga-ucsd.github.io

Education

M.S. in Electrical and Computer Engineering **University of California, San Diego**
GPA:3.5 out of 4.0 *— Coursework: Statistical Learning, Visual Learning, Linear Algebra* *Sep. 2023 – May 2025*

Bachelors in Electrical Engineering **Indian Institute of Technology, Ropar**
GPA:3.6 out of 4.0 *— Coursework: Machine Learning, Data Structures and Algorithms* *Jul. 2019 – Mar 2023*

Technical Skills

Languages: Python, C/C++, Assembly
Parallel & Distributed Computing: CUDA, CuPy, JAX, PyTorch Distributed, Ray, MPI, NCCL, torch.multiprocessing
Developer Tools: Git, Docker, Linux, Emacs, Vim
Libraries: PyTorch, TensorFlow, LangChain, StableBaselines, CuPy
Skills: Parallel Algorithms, Distributed Systems, High-Performance Computing (HPC)

Experience

Research Assistant – ***ERL Lab (Prof. Nikolay Atanasov)*** Apr 2024 – Present
UC San Diego

- Designed and deployed a scalable **parallel learning framework** to accelerate vision-based robotic grasping policies across multiple simulated environments.
- Engineered a **multi-process PyTorch pipeline** using Ray and torch.multiprocessing to vectorize CNN rollouts, and accelerated SAC training by offloading numerical kernels to **CuPy** and GPUs.
- Integrated **signed distance function (SDF)** models with **JAX** for auto-vectorized gradients, while containerizing multi-GPU workflows with Docker for reproducibility and portability.
- Produced a robust system capable of training transformer-based robotic policies at scale, thereby reducing experimental turnaround time and enabling large-scale benchmarking.

Research Assistant – ***MURO Lab (Prof. Sonia Martinez)*** Oct 2024 – Present
UC San Diego

- Developed a distributed backend infrastructure for **retrieval-augmented generation (RAG)** chatbots, focusing on scaling embeddings and inference workloads across heterogeneous clusters.
- Implemented **Ray Serve**-based microservices to parallelize embeddings and retrieval across CPUs/GPUs, and fine-tuned **Gemma 2.0 9B** with pipeline parallelism and gradient checkpointing.
- Optimized dense retrieval with **multiprocessing queues**, custom CUDA kernels, and memory-augmented RAG workflows, benchmarking against state-of-the-art frameworks like **vLLM** and TGI.
- Delivered a high-throughput distributed RAG system that achieved sub-second query latency, demonstrating scalability for production-level conversational AI.

Researcher – ISNL Lab, ***Jacobs School of Engineering*** Jan 2024 – May 2024
20hrs/week

- Developed and trained an **LSTM** model for brain intention detection using **EEG data** collected from a control group.
- Detected peaks corresponding to brain intentions in the data using **Fast Fourier Transform(FFT)** accelerated with **CuPy**.
- Scaled 500GB EEG data loading with **asynchronous batching**, decreasing the time to load data for the FFT analysis.
- Parallelized **EEG data decoding pipeline** using CPU Multi threading and with complete GPU time-series inference thus producing a low error prediction model for intention detection.

Data Scientist Intern – Samsung R&D Jun 2022 – Sept 2022
40 hrs/week

- Worked on **Big data** for Bixby Voice Assistant with the intent of improving user experience.
- Parallelized Bixby sentiment inference with **Ray** batch predictions across CPU cores hence decreasing the response time from 0.3ms to 0.1ms
- Optimized speech-text analytics with **multiprocessing + CuPy** acceleration, decreasing pipeline latency.
- Integrated the solutions in a team of 4 thus improving throughput for **large-scale audio processing**.

Research Assistant – ***Wireless Signal Processing Group (Prof. Satyam Agarwal)*** Dec 2021 – Feb 2022
IIT Ropar

- Explored deep learning approaches for **radio signal classification** with an emphasis on real-time inference under constrained hardware conditions.
- Implemented GPU-accelerated training pipelines in **TensorFlow** to process large volumes of RF data efficiently, leveraging optimized kernels for faster convergence.
- Streamed **GNURadio** signal data using multiprocessing, enabling continuous RF collection and integration with the classification pipeline for online learning.
- Delivered a practical proof-of-concept that showcased the feasibility of deploying GPU-based models for low-latency radio frequency classification in real-world systems.

Projects

Large-Scale Vision-Language Model Training | ***PyTorch, Ray, Docker, CUDA***

- Implemented distributed pretraining of a CLIP-style model on 20M image–text pairs using **PyTorch Distributed + NCCL** with mixed precision training.
- Parallelized data loading and augmentation pipelines with **Ray**, reducing GPU idle time by 35%.
- Benchmarked scaling efficiency on multi-GPU clusters, achieving >90% throughput compared to single-node baselines.

End-to-End Retrieval-Augmented Generation (RAG) System | ***Python, LangChain, Ray Serve***

- Developed a **chatbot with RAG** that integrates dense retrievers with fine-tuned LLMs for knowledge-grounded QA.
- Deployed on **multi-GPU clusters** with **Ray Serve**, sharding embedding + generation workloads for <1s query latency.
- Optimized retrieval pipelines with **Faiss + CuPy**, supporting 1M+ documents at production scale.

Reinforcement Learning for Robotic Control at Scale | ***PyTorch, JAX, CUDA, Docker***

- Trained **SAC/Transformer-based RL policies** for robotic grasping in MuJoCo using **GPU-parallelized SDF models**.
- Leveraged **JAX** auto-vectorization (`vmap`, `pmap`) for batched rollouts, improving training throughput by 3×.
- Containerized multi-node experiments with Docker + Ray, enabling reproducible large-scale benchmarking.

Scalable Generative Models for Image Synthesis | ***PyTorch, JAX, CUDA***

- Implemented **GAN and diffusion models** for high-resolution image generation, integrating **JAX** for distributed gradient computation.
- Optimized training with **mixed precision** and **CuPy**-accelerated preprocessing to fully utilize GPUs.
- Benchmarked generative quality with FID/IS metrics, improving over baseline implementations.