# Thomas Mbrice

US Citizen | mbricethomas@gmail.com | Github | Linkedin

## EDUCATION

**Stony Brook University** *Stony Brook, New York*
M.S./B.S. Computer Science & B.S. Applied Mathematics, *Expected May 2027*
*Coursework: Probability Theory, Natural Language Processing, Computer Networks*
Honors & Scholarships: Dean's List, Fall 2022, 2023, 2024, Presidential Merit Scholarship: 2022 - 2026

## TECHNICAL SKILLS

**ML/Systems:** PyTorch, distributed training (DDP/multi-GPU), CUDA, SLURM/HPC, NumPy, scikit-learn, sentence-transformers, HNSW/FAISS, RAG, DPO/LoRA fine-tuning, transformers
**Languages:** C, C++, Python, Bash, Java, TypeScript, R, SQL
**Infrastructure:** Docker, FastAPI, PostgreSQL, GCP, CI/CD, pytest, asyncio, Git
**Publications:** Stance Detection in Prediction Markets, Fair-Aurora RL Networks, LSTM Based Reserving Analysis

## EXPERIENCE

**Machine Learning & Data Science Intern** 06/2026 – 08/2026
*Dell Technologies* *Austin, TX*
- Built and deployed an encoder-based semantic search service over 2k-30k external documents, pairing mpnet embeddings with an HNSW index tuned to match brute-force recall@10 exactly while cutting p95 query latency 35x.
- Turned HNSW cluster density into a demand signal for sales and marketing, using semantic regeneration passes over the prompt pool and inverse-propensity weighting to keep clusters from mirroring query distribution.
- Refactored evidence function in internal search engine resulting in a 30% higher hit rate while searching 80% less tables.

**Graduate Researcher, ML for Quantum Systems** 02/2026 – Present
*Brookhaven National Laboratory (U.S. Dep. of Energy) & Stony Brook University* *Stony Brook, NY*
- Own ML engineering for a Brookhaven quantum physics team's DNN mapping source-subsystem density matrices to target subsystems in random quantum circuits.
- Cut inherited 200% relative error to 12% via ablation-driven loss design; plain MSE beat fidelity-constrained objectives whose physics constraint layers bottlenecked gradients.
- Cut A100 epoch wall-clock from 17h to 7min (145x) via parallel data loading, multi-GPU utilization, and a hierarchical rewrite that restricts each timestep to interacting subsystems.

**Lead ML Engineer** 08/2024 – Present
*Stony Brook University, Vertically Integrated Project* *Stony Brook, NY*
- Led four undergraduates building ethical decision-making agents for autonomous vehicles in CARLA, training RL policies under Bradley-Terry preference learning over 240 human-labeled pedestrian scenarios.
- Led a cross-disciplinary team of four undergraduate students in developing accurate modeling of autonomous vehicles in a 3D environment experimenting with KNN, Decision Trees (Random Forest) and Neural Networks.
- Built a SLURM-based data pipeline on SeaWulf HPC with automated validation, preprocessing, and augmentation, cutting experiment turnaround 60% and making runs reproducible across the team. | Github | VIP

## PROJECTS

**AEGIS - Fault Tolerance for Distributed Training** | Python, PyTorch, asyncio, pytest | Github 2026
- Built an in-process runtime composing three published recovery primitives, R²CCL (transport), MeCeFO (compute), TierCheck (storage), under one escalation controller.
- Designed a B0–B4 blast-radius taxonomy with a one-directional escalation invariant, verified under 60-fault chaos injection; 3+ correlated rack-level faults re-classify to full restore instead of silent neighbor-absorb, and every degraded fallback sets an explicit fidelity flag.
- Measured 92% reduction in $/GPU-hr lost versus a checkpoint-restart baseline under synthetic fault injection across 200 simulated ranks

**Akkadian Machine Translation,** PyTorch, ByT5, BM25, sentence-transformers | Github 2026
- Built byte-level seq2seq translation pipeline (ByT5 + BM25/genre-filtered RAG + lexicon integration) for 4,000-year-old cuneiform with 1,589 training pairs.
- Developed 8-stage diagnostic framework testing every pipeline component in isolation, catching bugs invisible to loss curves (self-retrieval leakage, scaffold-target mismatch, output truncation) improving genre precision from 21% to 92%.

**Fair-Aurora: Fairness Strategies for RL-Based Congestion Control**, Python, NS3 | arXiv | Github 2026
- Investigated fairness failures in Aurora (deep RL congestion controller) when deployed in multi-flow networks; implemented three post-hoc strategies: reward shaping, observation augmentation, and loss-sensitivity tuning.
- Found reward shaping achieved the best fairness-throughput tradeoff via Jain's fairness index; 3rd tested strategy emerged as most TCP-friendly under dynamic flow conditions.