Aashish Sheshadri

610-701-1740 | aashish.sheshadri@gmail.com | SF Bay Area, CA
github.com/aashish-sheshadri | linkedin.com/in/aashish-sheshadri | aashish-sheshadri.github.io

PROFILE

Twelve years building the machinery that decides whether an unreliable judgment is trustworthy enough to act on. First for human annotators, now for agents. It started with SQUARE, a benchmark for aggregating disagreeing human judges into a signal you could defend, and runs through **graduated autonomy**, the company-wide standard for how much autonomy an ML system is granted, to today's agentic SDLC platform: verifiable environments, programmatic graders, and authority scoped so no agent holds a write path. Individual contributor throughout — the standards I set were adopted by orgs I do not run.

5 peer-reviewed publications, 362 citations across all work (Google Scholar, Aug 2026), 1 issued patent and 1 pending as lead inventor, and AAAI HCOMP's Test of Time Award for SQUARE.

TECHNICAL SKILLS

Languages: Python, C++, Java, JavaScript; Go, C, SQL
Frameworks: PyTorch, TensorFlow, Kubernetes, GCP/AWS/Azure, Spark, Ray, Jupyter
Domains: Agent Harnesses & Tool-Use Safety, Grader & Reward Design, Verifiable Environments, Evaluation Methodology & LLM-as-Judge Reliability, Reinforcement Learning, Deployment Safety & Oversight, ML Infrastructure

EXPERIENCE

PayPal Inc, San Jose, CA **June 2014 – Present**

Machine Learning Architect (Sr. MTS) Feb 2023 – Present

Set ML and infrastructure strategy across cloud platform, data, and developer productivity: which problems get a model vs. a heuristic, and how production rollouts are governed across ~1,550 applications.

Agentic SDLC

- Architected the multi-agent platform that carries an engineer's stated intent to production, and built it with a team of four: provisioning, onboarding, ephemeral verifiable environments per change, programmatic gates, canary promotion. Specialist LLM agents sit behind one orchestrator on a shared harness, so memory, policy and a skill catalog are built once rather than per agent. In production across the infrastructure and application lifecycles; the data lifecycle agent has not shipped.
- Cut application onboarding from ~20 days of hand-held platform support to ~30 minutes of agent-driven provisioning. The predecessor was a self-serve model I had built and watched stall at that 20-day cost. This platform exists because the first system's limit was measured rather than assumed.
- Set the safety envelope: no agent holds a write path, and every mutation is executed by deterministic code acting on an agent's proposal. Two independent points enforce it. A pre-tool-use hook denies destructive changes before the call is issued, and policy-as-code re-validates at apply time, so the policy holds regardless of which agent, or which human, merged. Authority is budgeted by blast radius: an open-ended tool loop for the agent furthest from production, a short schema-validated pipeline for the one that touches it.
- Designed the refusal path so an agent can act on it. A plan that exceeds its resource cap comes back as a structured policy verdict, not a stack trace; the agent reads the verdict, corrects the change, and re-submits without a human in the loop.
- Made promotion a graded decision rather than a review: an evidence-first grader scores each change against its acceptance criteria and a state transition keyed on that verdict holds it in draft until it passes. Closed the loop to an independently built code agent through a neutral check-run contract, so either side can add a gate without a coordinated deploy. Grading against real environments caught what static policy checks could not: a cross-environment permission gap and an identity-binding mismatch, both before they reached customers. No aggregate false-pass rate yet; the gate has been live too briefly to quote one.

- Guarded the shared agent memory against drift: agents compose only from canonical pre-approved references, and a dedicated review skill validates every change before it is proposed, so each run inherits the policy failures the last one hit.

ML systems and deployment governance

- Designed the ML deployment framework (canary evaluation, shadow mode, **graduated autonomy**), now the company-wide production standard governing how much autonomy an ML system is granted, including the eval bar a system must clear at each step before its autonomy widens. Introduced as a Staff engineer.
- Set autoscaling and capacity strategy, and built the hierarchical RL policy for cluster capacity allocation across ~1,550 applications (~100k pods), acting on the demand forecasts I had built for the same fleet. The hard part was reward function and curriculum engineering, not the policy architecture.
- Found and closed reward-specification failures under optimization pressure, cases where the policy optimized the written objective into an outcome I had not intended. Optimizer-found, not model-reasoned: the policy is small and non-linguistic and cannot represent its own evaluation.
- Ongoing collaboration with IBM Quantum on variational algorithms for risk modeling (published in *Quantum*, 2023).

Staff Machine Learning Engineer (MTS 2) Mar 2020 – Feb 2023

- Led ML-driven predictive autoscaling for ~1,500 applications: an automated policy given authority to change production capacity, with five-nines availability as the binding constraint. Earning enough confidence to let it act was the work; the ~25% efficiency gain followed from it.
- Carried the sequence-forecasting work from infrastructure signals to demand: time-series models with confidence-calibrated predictions for ~1,500 applications, handling distribution shift across seasonal, promotional, and organic traffic patterns. Those forecasts became the autoscaler's input signal.
- Developed sequence classification models using BERT for real-time threat detection in information security, security-critical inference where false negatives are expensive.

Senior Machine Learning Engineer (MTS 1) Mar 2018 – Feb 2020

- Applied sequence-to-sequence models (RNN/LSTM) to CPU and memory forecasting, replacing hand-tuned static alert thresholds with likelihood-based outlier detection, and scaled it across the ~350,000-signal monitoring surface. Presented the approach at QCon.ai 2019 (infoq.com/presentations/journey-99999).
- Built the research-compute platform underneath it: multi-tenant accelerator scheduling under contention with prioritized access, remote kernel orchestration, and reproducible isolated environments from experiment through production serving.
- Built the AST-level parameter substitution behind PPExtensions' `run_pipeline`, which type-checks and injects values into each notebook's parameter cell so parameterized stages receive state. Plus Airflow scheduling and the CSV, logging and config layers (github.com/paypal/PPExtensions).

Senior Research Engineer Mar 2016 – Feb 2018

- Co-authored the FASH64 specification with Doug Crockford (JSON inventor): a fast non-cryptographic hash, validated by statistical dispersion analysis and empirical collision-rate testing (crockford.com/fash64.pdf). Designed and open-sourced the **Seif Protocol**, an ECC-512 handshake over Node.js (github.com/paypal/seif-protocol).
- Research in probabilistic analysis and randomness validation, the distribution-modeling and uncertainty-quantification toolkit the later ML work runs on.

Research Engineer Jun 2014 – Feb 2016

- Built the cryptographic foundation under Doug Crockford: **seifnode**, a Node.js addon over ISAAC RNG, ECC, AES-GCM and SHA3-256, and the C++ entropy-mining CSPRNG beneath it. 543 stars, MIT (github.com/paypal/seifnode). Presented at OSCON, 2015 and 2016.

The University of Texas at Austin, Austin, TX **Sept 2012 – May 2014**

Graduate Research Assistant

- Judge reliability and evaluation quality: aggregating noisy, disagreeing judgments into a defensible signal, and hybrid expert/non-expert judging for building test collections affordably. These are the problems now called annotation quality and LLM-as-judge reliability.
- Built and open-sourced the SQUARE benchmark (MIT), a common harness for comparing consensus algorithms (majority vote, GLAD, CUBAM, Dawid-Skene) under supervised, semi-supervised, and unsupervised regimes, with a worker-simulation framework for generating judges of known quality: github.com/utir/square and github.com/utir/square-2.0. **AAAI HCOMP Test of Time Award, Oct 2024**; 267 citations.

Teaching Assistant Sept 2012 – Dec 2012

- Programming Languages (undergraduate course)

Carnegie Mellon University, Pittsburgh, PA **Sept 2011 – May 2012**

Research Associate

- LiDAR-camera calibration for mapping and localization. Visual odometry and orthographic views for global localization of planetary rovers. Contributed to Astrobotic's Google Lunar XPRIZE effort.

EDUCATION

The University of Texas at Austin — M.S. Computer Science, GPA 3.90 Aug 2012 – May 2014

Computer Vision & IR Thesis Track

Carnegie Mellon University — Audited Courses in Computer Vision, Robotics & Perception Aug 2011 – May 2012

PES Institute of Technology, Bangalore — B.S. Electronics & Communication, GPA 8.40, Honors Jun 2007 – Aug 2011

PUBLICATIONS

- Variational quantum algorithm for unconstrained black box binary optimization: Application to feature selection — *Quantum*, Jan 2023 (arxiv.org/abs/2205.03045)
- Mix and match: collaborative expert-crowd judging for building test collections accurately and affordably — *DESIRES*, Aug 2018
- A collaborative approach to IR evaluation (Masters Thesis) — *UT Digital Repository*, May 2014
- SQUARE: A Benchmark for Research on Computing Crowd Consensus (First Author, **Test of Time Award, Oct 2024**) — *AAAI HCOMP*, Nov 2013. Code: github.com/utir/square
- Position Estimation by Registration to Planetary Terrain (First Author) — *IEEE MFI*, Sept 2012
- Complementary Flyover and Rover Sensing for Superior Modeling of Planetary Features — *FSR*, July 2012

PATENTS

- **Session Management System** (Lead Inventor) — US11818219, Issued Nov 14, 2023. Embedding-based behavioral modeling for security: session logs vectorized against a dictionary spanning both natural and formal language, building a per-user model that flags deviation as anomalous.
- **Dynamic Autoscaling of Server Resources Using Intelligent Demand Analytic Systems** (Lead Inventor) — application US 17/826,071, filed May 26, 2022 (pending)

INVITED SPEAKER & PEER REVIEW

QCon, Strata, MLConf, Deep Learning World, RE•WORK Enterprise AI Summit, OSCON, Fluent. Recorded: *On a Deep Journey towards Five Nines* — QCon.ai 2019, infoq.com/presentations/journey-99999

Peer reviewer: Information Processing & Management (IF 7.4), Artificial Intelligence (IF 5.1)