Haichuan Zhou

Los Angeles, CA | 13812764054zhc@gmail.com | +1 (934) 799-8458 | LinkedIn | GitHub

AI Engineer building grounded agent systems for codebase understanding, retrieval, and evaluation: LangGraph supervisor workflows, self-authored MCP servers, production FastAPI/cloud infrastructure, and eval-driven anti-hallucination loops.

EDUCATION

University of Southern California - M.S. in Analytics (STEM), Los Angeles, CA | Expected Dec 2026
Coursework: Machine Learning for Analytics, Big Data Management, Data Mining, Statistical Learning

New York University - B.S. in Mathematics, Minor in Computer Science, New York, NY | May 2025
Coursework: Data Structures and Algorithms, Design and Analysis of Algorithms, OOP, Database Systems, Applied Probability

TECHNICAL SKILLS

AI / Agents: LangGraph, LangChain, MCP, FastMCP, HITL workflows, agent verification, RAG, LLM-as-judge evals, OpenAI API, Anthropic Claude API
ML / Modeling: PyTorch, transformer/BERT embeddings, recommendation & ranking models, embedding-based retrieval
Code Intelligence: LibCST AST/CST indexing, static dependency graphs, symbol grounding, call-chain extraction, test-execution verification
Backend / Infra: Python, FastAPI, Pydantic v2, SQLAlchemy 2.0 async, PostgreSQL, pgvector, Redis, Docker, AWS Lambda/SQS/DynamoDB/ECS, Terraform
Quality / Observability: pytest, Jest, strict mypy, ruff, GitHub Actions, OpenTelemetry, LangSmith, Langfuse, CloudWatch, k6

PROJECTS

wayfinder - Multi-agent codebase onboarding copilot with embedded verification
LangGraph Supervisor, FastAPI, Next.js, OpenTelemetry, 5 MCP servers, Cloud Run | GitHub

- Built a deployed repo-onboarding agent that routes GitHub repository questions across architect_mapper, entry_explainer, and verifier agents; integrates 5 MCP servers and labels high-risk code claims as verified, unverified, or contradicted before final output.
- Hardened natural-language-to-symbol grounding so symbol, CLI, and behavioral questions on real repos return verified file:line citations — resolving backticked/dotted symbols, pyproject CLI entry points, and module source — with ambiguous names kept unverified rather than guessed.

MCP Codebase Tools - Three self-authored MCP servers for grounded code understanding
FastMCP, LibCST, pytest/Jest, PyPI, Official MCP Registry | repo-mapper | ast-explorer | test-runner

- Authored and shipped 3 independent MCP servers that give onboarding agents deterministic structure, semantic, and verification facts before any LLM explanation step; published all three as public GitHub repos, PyPI packages, and Official MCP Registry records.
- Implemented repo scanning, framework detection, entry-point discovery, LibCST-backed definitions/references/signatures/call chains/class hierarchy, and bounded pytest/Jest execution with timeouts, cwd restrictions, normalized results, and coverage summaries.

agent-eval-harness - Open-source framework for agent regression testing
Python CLI, LLM-as-judge, golden-set CI, failure-mode taxonomy, LangGraph/LangChain adapters | GitHub

- Built an open-source framework scoring LLM agents on four metrics — routing accuracy, factual correctness (LLM-as-judge with self-consistency), citation grounding, and test-execution verification rate — architecture-blind scoring, green under ruff + mypy --strict + 73 pytest tests.
- Shipped a Supervisor-vs-ReAct benchmark over 40 tasks across 10 Python OSS repos (model and 5 MCP tools fixed): the Supervisor used ~12x fewer tokens (396K vs 4.8M) and finished all 40, while ReAct failed 6/40 on recursion-limit blowups.

Auto-Apply Agent - LangGraph pipeline that scores and auto-fills job applications across any standard ATS
LangGraph (ingest / jobs / apply subgraphs), browser-use, ChromaDB, JobSpy, Anthropic SDK, SQLite, Chart.js | HireBeat (internal)

- Built one LangGraph pipeline (modular ingest / jobs / apply subgraphs) that parses a resume into a ChromaDB knowledge base, scores each job against it, and auto-fills applications on any standard ATS (Lever / Greenhouse / Ashby / Workday); live-validated end-to-end on a 30-job concurrent batch.
- Designed a multi-dimension JD match-scoring service — four weighted dimensions (Hard Skills 40 / Experience 30 / Soft 10 / Must-haves 20) scored against ChromaDB-retrieved resume evidence, with a hard veto on must-haves (degree, work authorization) and a >80 apply threshold; validated on 49 real scraped jobs (12 qualified).
- Engineered a browser-use form-filler with semantic field mapping, per-site domain-locking, and a safe shadow-click design (never submits), plus type-aware human-in-the-loop takeover for CAPTCHAs and login walls — validated live on reCAPTCHA, GeeTest slider, and Cloudflare Turnstile; runs DOM/AOM-only (no screenshots) to cut token cost.

arxiv-rag - Production Retrieval-Augmented Generation system over 50K arXiv abstracts
PostgreSQL pgvector, BM25 + dense retrieval, reciprocal rank fusion, Cohere rerank, RAGAS, Railway | GitHub

- Built and deployed a hybrid-retrieval RAG service over 50K arXiv abstracts (dense + BM25 with reciprocal rank fusion, Cohere rerank, pgvector HNSW, inline citations); ran a 256/512/1024-token chunk ablation and published the counter-intuitive 1024-token win as a bilingual article.

EXPERIENCE

HireBeat Inc. - AI Engineer Intern, Jersey City, NJ | Jun 2026 – Present

- Sole engineer on a ~2.9K-LOC autonomous job-application agent (LangGraph + Anthropic Claude): built a grounded, anti-hallucination scoring engine that decomposes each JD into four weighted dimensions and scores each only against top-k=6 ChromaDB-retrieved résumé evidence, computes the weighted total deterministically in Python (not the LLM), and enforces a hard degree / work-authorization veto that can override a high score; validated on 49 scraped jobs (12 qualified).

CampusBridge - Machine Learning / Backend Engineer Intern, Remote | Jan 2025 - Sep 2025

- Trained a neural student–job matching / recommendation model in PyTorch: encoded résumés and job postings into transformer-based (BERT) embeddings and learned a ranking model to score candidate–role fit for the placement platform.
- Built the data pipeline behind the model — automated ingestion and normalization that turned messy multi-source records into clean, schema-consistent training and feature data, cutting manual cleanup across profile and placement data.
- Designed the relational schemas and REST APIs serving the matching, tagging, and résumé-generation workflows; optimized large-dataset retrieval on hot matching paths via indexing and query tuning to keep candidate-scoring queries fast at scale.