KUMAR SATVIK CHAUDHARY

Tempe, AZ | kchaud13@asu.edu | LinkedIn | GitHub | Google Scholar

EDUCATION

Arizona State University, Tempe, AZ

Master of Science in Computer Science (Accelerated) **(GPA: 4.0/4.0)** **Expected May 2027**

Bachelor of Science in Computer Science, Minor in Data Science **(GPA: 4.0/4.0)** **May 2026**

WORK EXPERIENCE

ML & NLP Research Intern (REU), Arizona State University, Tempe, AZ **Jan 2025 – Present**

- Developed an **LLM-powered cybersecurity tutoring platform** integrating FAISS-based retrieval, intent classification, and ontology-grounded response validation; deployed to **150+ students** and earned the **CIKM 2025 Best Demo Paper Award**.
- Trained **PyTorch and Hugging Face transformer models** on **24K+ essays**, implementing 8 rubric-based concept predictors and joint optimization of concept scores and final grades to provide interpretable writing feedback.
- Contributed to **rubric-aware concept models** with ordinal calibration and latent error correction; evaluated across **3 datasets and 5 transformer backbones**, outperforming **4 CBM baselines in all 15 task-F1 comparisons**.
- Built a **FastAPI backend and Streamlit application** for interactive essay grading, allowing users to modify rubric scores and recalculate predictions without rerunning the transformer encoder.

Software Engineer (Capstone), Arizona State University, Tempe, AZ **Aug 2025 – May 2026**

- Built and deployed a **full-stack RISC-V emulation platform** using React, TypeScript, and Python, eliminating local VM/IDE setup through browser-based access on Vercel.
- Developed an **RV32I execution engine** supporting **40+ instructions**, single-step debugging, register/memory inspection, and simulated hardware I/O for interactive assembly programming.
- Built **Python REST APIs** and PostgreSQL-backed workflows for assignment management, code submissions, and grading, with user authentication and instructor/student access controls.

PROJECTS

Footnote: Citation-Grounded RAG System | GitHub| Demo Video *Python, FastAPI, React, TypeScript, SQLite, Docker*

- Built a **full-stack RAG platform** over **6 document formats**, streaming answers via **Server-Sent Events** with **sentence-level citations** linking each claim to its exact source passage, section, and page.
- Engineered **hybrid retrieval** fusing **BM25** and **dense embeddings** via **Reciprocal Rank Fusion**, outperforming BM25-only and dense-only on 61 labelled questions (**MRR 0.74** vs. 0.71 and 0.64; **Recall@5 86%**).
- Mitigated **LLM hallucinations** via a **cross-encoder relevance gate** that refuses unanswerable questions before any LLM call (**92% balanced accuracy**), citation-enforced prompting, and a **post-generation verifier**.
- Designed an **evaluation harness** (Recall@k, MRR, nDCG, **LLM-as-judge** faithfulness) with chunking-invariant labels; shipped with **75+ automated tests**, Docker, and a **GitHub Actions CI** pipeline.

Tally: Self-Correcting Text-to-SQL Agent | GitHub *Python, LangGraph, FastAPI, React, TypeScript, SQLite*

- Built a **self-correcting text-to-SQL agent** as a **LangGraph** state machine that links the schema, generates and validates SQL, executes it, and feeds database errors back to the LLM for bounded repairs, streaming each step via **SSE**.
- Engineered **defense-in-depth SQL guardrails**: an **sqlglot AST allow-list**, read-only SQLite connections, an **authorizer callback**, and row/time/size budgets, each tested in isolation to block writes, PRAGMA, ATTACH, and runaway queries.
- Implemented **BM25 schema linking** with value matching and **foreign-key join-path expansion**, passing only **2.9 of 11 tables** per question while retaining **100% of required tables** on a **51-question** execution-accuracy benchmark.
- Added a **bring-your-own-data library** (CSV, Excel, SQLite; untrusted SQLite rebuilt into fresh files) with relationship inference, **grounded LLM insights** checked against result rows, 6 chart types, and **300 automated tests** in CI.

Voot: AI-Powered Accounts Payable Automation *Python, FastAPI, React, TypeScript, PostgreSQL, Azure, Docker*

- Developed a **full-stack accounts payable platform** with **7-stage document lifecycle tracking**, role-based access control, invoice review workflows, and organization-scoped financial records using React, FastAPI, and PostgreSQL.
- Implemented an **asynchronous invoice processing pipeline** across **5 Docker services**, integrating Azure Blob Storage, Storage Queues, Python workers, and a configurable Azure Document Intelligence extraction provider.
- Added **5 financial intelligence capabilities**: invoice validation, duplicate detection, review-risk scoring, GL account suggestions, and vendor correction-email generation, with structured extraction and audit logging.
- Delivered **REST APIs, financial dashboards, CSV exports, and document reprocessing**; verified **23 automated tests** (21 backend, 2 frontend), PostgreSQL migrations, and end-to-end upload and duplicate-detection workflows.

SKILLS

Programming Languages: Python, C++, R, SQL
ML & AI: Scikit-learn, TensorFlow, PyTorch, Hugging Face Transformers, OpenAI APIs, MLflow
Data Science & Analytics: NumPy, Pandas, Matplotlib, Seaborn, Plotly, Power BI, Excel, SHAP
Frameworks & Developer Tools: FastAPI, Flask, LangChain, RAG, REST APIs, Docker, Git/GitHub, Linux, Jupyter
Databases: MySQL, PostgreSQL, MongoDB