SATHWIK NAAG SANKURU

AI/ML ENGINEER | AGENTIC AI | RAG | DEEP LEARNING

Location: Hyderabad, Telangana, India
Mail: sathwiknaag12@gmail.com
Phone: +91 8374043278
LinkedIn: https://www.linkedin.com/in/sathwik-naag-sankuru-820756386
Portfolio: https://sathops.github.io/SathwikNaagPortfolio/
GitHub: https://github.com/SathOps

PROFILE

Final-year B.Tech Computer Science (Data Science) student focused on Agentic AI, Retrieval-Augmented Generation (RAG), and Deep Learning, with hands-on experience building agentic AI systems, retrieval pipelines, and end-to-end ML applications in Python. Experienced with LangChain, ChromaDB, Hugging Face embeddings, LLM integration, structured outputs, Transformers, CNNs, RNNs, and Explainable AI. Strong foundation in developing AI pipelines involving data preprocessing, retrieval, model training, evaluation, optimization, and inference. Interested in building reliable, scalable AI systems that combine reasoning, perception, and autonomous execution for real-world applications.

EDUCATION

GITAM University Hyderabad (2023 - 2027) | B.Tech - CSE (Data Science) **CGPA : 7.72**

CORE SKILLS

Programming: Python | Java | JavaScript | SQL

AI & Machine Learning: Machine Learning | Deep Learning | Neural Networks | Transformers | CNN | RNN | Computer Vision | Feature Engineering | Data Preprocessing | Model Training | Model Evaluation | Explainable AI

Generative AI & LLMs: RAG | Agentic AI | LLM Applications | LangChain | Hugging Face | Sentence Transformers | Prompt Engineering | Structured Outputs | Pydantic | OpenAI | Google Gemini | Ollama | ChromaDB | Vector Retrieval

Data & Application Development: Exploratory Data Analysis | Data Visualization | Streamlit | REST APIs | MySQL | TensorFlow | OpenCV

Computer Science & Tools: Data Structures & Algorithms | OOP | DBMS | Git | GitHub | GitLab | Problem Solving

EXPERIENCE

AI & Full Stack Developer Intern | Swecha.ai (May 2026 – Jun 2026)

- Built a Retrieval-Augmented Generation (RAG) chatbot for document-grounded question answering using Python and LLM-based information retrieval.
- Worked with document processing, retrieval, context injection, and response generation to produce grounded answers.
- Collaborated with developers, designers, and stakeholders to translate requirements into production-ready digital experiences.
- Led the UI/UX redesign of the official Activity Tracker platform, improving usability, navigation, information architecture, and workflow efficiency.
- Redesigned the official Swecha website with a focus on accessibility, responsive design, visual hierarchy, and user engagement.
- Selected Contributions & Project Repositories: GitLab Link

AI/ML Intern – Iris Flower Classification Project | BITS Pilani – Hyderabad Campus (7-Week AI/ML Internship)

- Developed a Logistic Regression classification pipeline using Python and Scikit-learn to classify Iris flower species.
- Performed data preprocessing, feature analysis, model training, and evaluation using standard classification metrics.
- Built a reusable inference workflow by serializing the trained model using Joblib.
- Tech Stack: Python, Scikit-learn, Pandas, NumPy, Joblib | Google Colab Link

PROJECTS

Legal Intelligence RAG Agent — Agentic AI System

- Developed a single-agent legal intelligence system using a central **LegalResearchAgent** to orchestrate intent classification, evidence retrieval, LLM reasoning, structured output generation, and schema validation.
- Implemented LLM-driven tool calling with LangChain, enabling dynamic selection and execution of tools for case-evidence retrieval, timeline extraction, witness contradiction analysis, and case-fact summarization.
- Built a RAG pipeline using ChromaDB and Sentence-Transformer/Hugging Face embeddings to retrieve and relevance-score domain-specific legal evidence.
- Integrated Ollama, Gemini, and OpenAI through a unified LLM service with provider fallback handling and support for local/offline inference using Ollama.
- Enforced structured responses with Pydantic validation, incorporating grounded answers, key findings, evidence, limitations, and citations, with session history and tool visualization in Streamlit.
- **Future Scope:** Extending the single-agent architecture toward specialized agents for legal research, evidence retrieval, contradiction analysis, and timeline reasoning.

Tech Stack: Python, LangChain, RAG, ChromaDB, Hugging Face, Sentence Transformers, OpenAI, Gemini, Ollama, Pydantic, Streamlit | Github Link

Optimized Hybrid Deep Transformer Model for Enhanced Cyberattack Detection — Final Year Project (In Progress)

- Implemented and analyzed the base hybrid IDS architecture referred to in the research study, using the CICIoT2023 dataset for multi-class IoT cyberattack detection.
- Built an end-to-end preprocessing pipeline for large-scale network traffic data, incorporating streaming data ingestion, data cleaning, Log1p transformation, normalization, and feature preparation.
- Applied Binary Particle Swarm Optimization (BPSO) to identify informative network features, reducing the feature space from 46 to 27 features (41.3% reduction).
- Implemented and evaluated CNN-Transformer, XGBoost, and MLP models, followed by validation-driven probability fusion to establish a reproducible baseline hybrid IDS.
- The baseline hybrid system achieved **77.83%** accuracy, **0.7184** Macro F1, and **0.7985** Weighted F1 on the untouched CICIoT2023 test set.
- **Research Extension:** Currently designing and experimenting with an enhanced hybrid architecture and optimization methodology based on the baseline findings, with the objective of achieving improved detection performance, stronger generalization, greater feature efficiency, and reduced false positives.
- Conducting comparative experiments and ablation studies to identify the architectural and optimization strategies that provide meaningful improvements over the baseline model.

Tech Stack: Python, TensorFlow/Keras, CNN, Transformers, XGBoost, MLP, BPSO, Scikit-learn, CICIoT2023, Deep Learning, Feature Engineering

Traffic Sign Recognition — CNN Computer Vision System

- Developed a CNN-based image classification system to recognize **43** traffic sign categories using the GTSRB dataset.
- Built an end-to-end computer vision pipeline covering image preprocessing, **32×32** resizing, pixel normalization, categorical encoding, train-test splitting, model training, and inference.
- Designed a CNN architecture with convolutional and max-pooling layers, followed by a dense classification layer with Dropout regularization.
- Trained the model using the Adam optimizer and categorical cross-entropy, achieving **99.66%** test accuracy after **10** epochs.
- Generated a 43-class confusion matrix to visualize prediction errors and analyze classification performance across traffic sign categories.
- Applied TensorFlow/Keras, OpenCV, NumPy, and Scikit-learn for deep learning, image processing, data preparation, and model evaluation.

Tech Stack: Python, TensorFlow, CNN, OpenCV, NumPy | Google Colab Link