SANIL MISHRA

+91 8102240182 | Noida, India | sanilmishra010@gmail.com | sanilmishra.me | github.com/sanil2050

PROFESSIONAL SUMMARY

AI Engineer focused on Agentic AI, automation, and rapid MVP development. Experienced in building enterprise-grade workflows using multi-agent orchestration, hybrid retrieval-augmented generation (RAG), and LLM-powered automation. Comfortable working independently across the full stack — data pipelines, model training, ontology design, and backend APIs — and skilled at using AI coding agents (Claude Code, Gemini, Codex) to move from idea to working prototype in days, not weeks.

SKILLS

Agentic AI & LLM Tooling: LangChain, MCP (Model Context Protocol), Multi-Agent Orchestration, Hybrid RAG (Dense + Sparse Retrieval), Embeddings, Prompt Engineering, Claude Code, Gemini, Codex, GLM, Kimi, OpenClaw, Local Models

Programming Languages: Python, C/C++ (Intermediate), JavaScript (Beginner)

Machine Learning & AI: Machine Learning, Deep Learning, Generative AI, Model Training, Explainable AI (SHAP), Scikit-learn, XGBoost, PyTorch

Vector Search & Databases: Pinecone DB, SQL

Backend & APIs: Django, FastAPI, REST API, HTML, CSS, UI/UX

Data Analysis & Visualization: Pandas, NumPy, Microsoft Excel, Tableau, Power BI

Development Tools: Git, GitHub, VS Code, Code::Blocks, Eclipse

Soft Skills: Leadership, Teamwork, Communication, Planning, Time Management

CERTIFICATES

- Oracle Agentic AI Foundations | MITS: FDP on Agentic AI
- Oracle Cloud Infrastructure AI Foundations
- Microsoft Excel 2026: Beginner to Excel Pro in only 6 Hours
- Data Science Mastery 2025: Excel, Python & Tableau

TECHNICAL EXPERIENCE

Enterprise RAG Platform **2026**

Python, FastAPI, LangChain, PostgreSQL + pgvector, Google Gemini, Nemotron, JWT/RBAC | github.com/sanil2050/enterprise-rag

- Built a production-oriented RAG platform for secure question-answering over internal company knowledge, enforcing JWT authentication and RBAC-based permission filtering so retrieval only surfaces documents the requesting user is authorized to access
- Designed a routing pipeline that rewrites incoming queries using conversation history, then dispatches an Agent Planner across Document RAG (hybrid dense + sparse retrieval) and GraphRAG (graph-based retrieval) branches, with agent execution bounded to a maximum of 3 steps
- Fused results from both retrieval branches using Reciprocal Rank Fusion (RRF) and applied cross-encoder reranking to surface the most relevant evidence before generation
- Enforced an evidence-before-answers design — validating citations against retrieved evidence and generating grounded responses with Nemotron so every factual answer is traceable to a source document
- Built a document ingestion pipeline (PDF/DOCX extraction, page-aware chunking, embedding generation) storing vectors in PostgreSQL with pgvector, and exposed the system through a FastAPI server with interactive Swagger/OpenAPI docs

Hybrid RAG Pipeline — Agentic Retrieval-Augmented Generation System **2026**

Python, LangChain, Pinecone, HuggingFace Embeddings, Google Gemini | github.com/sanil2050/hybrid-rag-pipeline

- Ingested PDFs with pypdf, extracted and cleaned page-level text, then split it into ~500-character chunks (50-character overlap) using RecursiveCharacterTextSplitter for optimal retrieval granularity
- Generated dense vector embeddings with HuggingFace's all-MiniLM-L6-v2 (384-dim) and fit a BM25 sparse encoder, combining both into a single Pinecone hybrid index (dot-product similarity, serverless AWS deployment)
- Implemented batched upserts (100 vectors/batch) to populate the vector store, then built a PineconeHybridSearchRetriever (top-k = 5) that blends dense semantic similarity with sparse keyword relevance in one query
- Combined retrieved chunks into a grounded prompt and passed it to Google Gemini 2.5 Flash (temperature 0.2) to generate answers strictly from retrieved context, reducing hallucination risk
- Tracked source file name and page number through every retrieval step, making each generated answer traceable back to its origin document

Multi-Agent AI Research System **2026**

Python, LangChain, Google Gemini, Tavily API, Streamlit | github.com/sanil2050/Multi-Agent-Ai-Research-System

- Research Agent queries the Tavily Search API for a given topic to gather ranked titles, URLs, and snippet summaries from live web sources
- Reader Agent selects the most relevant URL and scrapes the page with BeautifulSoup, stripping navigation, scripts, and footers to isolate clean, usable text
- Writer Agent synthesizes the search snippets and scraped content into a structured report — Introduction, Key Findings, Conclusion, and Data Citations — using Gemini 2.5 Flash
- Critic Agent audits the generated report and returns a numerical score (1–10) along with strengths, improvement areas, and a final verdict, closing the feedback loop
- Sequenced all four agents in a LangChain-orchestrated pipeline (pipeline.py), exposed through a Streamlit web UI (live logs, expandable research data, downloadable reports) and a headless CLI mode for scripted runs

Breast Cancer Classification using YOLOv11 & YOLOv26 **2026**

Python, PyTorch, Ultralytics YOLOv11/YOLOv26, CBIS-DDSM, VinDr-Mammo

- Preprocessed and combined two public mammography datasets (CBIS-DDSM and VinDr-Mammo) into a unified classification dataset spanning benign and malignant cases
- Split the combined dataset into 70:10:10 train/validation/test sets and applied data augmentation (rotation, flipping, contrast adjustment) to improve generalization across imaging protocols
- Addressed class imbalance between benign and malignant samples using class-balancing techniques prior to training
- Ran systematic hyperparameter tuning (learning rate, batch size, image resolution) independently on YOLOv11 and YOLOv26 classification heads
- Benchmarked both architectures on held-out test data using precision, recall, and accuracy to compare diagnostic classification performance — scoped as a research study without a deployment stage

Object Detection using YOLOv8 **2025**

Python, PyTorch, Ultralytics YOLOv8

- Curated and labeled a custom dataset of cold-drink bottles with bounding-box annotations for supervised training
- Applied data augmentation (flips, color jitter, mosaic) and transfer learning from pretrained COCO weights to speed up convergence
- Trained YOLOv8 to 92% detection accuracy, then built an inference pipeline with configurable confidence and NMS thresholds
- Exported the trained model to ONNX format for lightweight, framework-agnostic deployment
- Evaluated detection quality across classes using precision, recall, and confusion matrix analysis

Hate Speech Detection Using Machine Learning **2025**

Python, Scikit-learn

- Collected and cleaned a labeled tweet dataset spanning hate speech, offensive language, and neutral categories
- Preprocessed text (lowercasing, punctuation and stopword removal) and vectorized it with CountVectorizer for a bag-of-words representation
- Trained a DecisionTreeClassifier for multi-class text classification and evaluated performance on a held-out test split
- Structured the pipeline end-to-end from raw text ingestion to prediction, making it easy to swap in alternative classifiers for comparison

Data Analysis Project **2025**

Python, Pandas

- Built a data analysis pipeline in Python/pandas to clean, transform, and explore a tabular dataset
- Generated visualizations and computed statistical summaries to identify trends and patterns for insight extraction

EDUCATION

Master of Computer Applications (MCA), Galgotias University

Bachelor of Computer Applications (BCA), Chandigarh University

Senior Secondary, T.P.S College

Matriculation, Dr. D. Ram DAV Public School

LANGUAGES

English, Hindi