

Muhammad Maroof

AI/ML Backend Engineer | Independent Researcher

Karachi, Pakistan | maroof96965@gmail.com | linkedin.com/in/maroofiums | github.com/maroofiums |
kaggle.com/maroofiums

PROFESSIONAL SUMMARY

AI/ML engineer who builds and deploys AI-powered backend systems. Hands-on experience with FastAPI and Django services, RAG pipelines, LLM fine-tuning, and CPU-efficient inference, including a text-to-SQL model made about 6.6x faster and 4.2x smaller with higher accuracy. Completed 3 internships, published 2 research papers on Zenodo, and ranked 335 on Kaggle. Currently a software engineering intern at Lucky Core Industries and a 3rd-year DAE Software Technology student at MITI Karachi.

TECHNICAL SKILLS

Languages: Python, SQL, TypeScript, C#

ML / AI: PyTorch, Scikit-learn, Hugging Face Transformers, LangChain, RAG, LLM fine-tuning (QLoRA), Prompt Engineering, Quantization, NLP, Feature Engineering, Model Evaluation (Accuracy, F1, ROC-AUC)

Backend: FastAPI, Django REST Framework, REST API Design, PostgreSQL, SQLAlchemy, Pydantic, NDJSON Streaming

Deployment / Tools: Docker, Git, GitHub, Streamlit, llama.cpp (GGUF), Pytest, Kaggle, Google Colab, React (working knowledge)

PROFESSIONAL EXPERIENCE

Software Engineering Intern

Lucky Core Industries (LCI)

Sep 2026 – Present

Karachi, Pakistan

- First assignment: build an AI agent for the company's internal .NET system to fetch information.
- Constraint: CPU-only infrastructure with no GPU. Compared three options (Microsoft Copilot Agent, local model with optimizations, hosted LLM API) and chose a local model with CPU optimizations.
- Building a CPU-optimized LLM applying quantization and CPU inference optimizations.
- Developing Retrieval-Augmented Generation (RAG) pipelines to answer questions from internal company data.
- Developing a voice-to-text (speech transcription) component for internal tooling.

ML Specialist Intern

Elevvo Pathway

Feb 2026 – Apr 2026

Remote

- Cleaned and preprocessed raw datasets, handled missing values, and performed exploratory data analysis to guide modeling choices.
- Engineered features and trained Scikit-learn classification and regression models.
- Evaluated models with Accuracy, F1-score, and ROC-AUC, and improved them through hyperparameter tuning.
- Documented every experiment and presented results and trade-offs to senior developers.

AI & ML Engineer

DeveloperHub Corporation

Dec 2025 – Feb 2026

Remote

- Built backend services with FastAPI and Django REST Framework.
- Developed LLM-powered chatbots using Hugging Face and OpenAI APIs, with prompt engineering and safety filters.
- Built a house price prediction model and deployed it as a Streamlit application.
- Containerized applications with Docker for reproducible deployment.

RESEARCH AND PUBLICATIONS

QuerySmith: Fine-Tuning Phi-4-mini for Text-to-SQL with CPU-Efficient Partial-Residual Quantization

Sep 2026

Zenodo preprint, single author (DOI 10.5281/zenodo.23000490)

- **Problem:** text-to-SQL models are usually built and evaluated for GPU inference, which blocks use in on-premise or resource-constrained settings with CPU-only hardware.
- **Fine-tuning:** Phi-4-mini-instruct (3.8B parameters) trained with QLoRA (rank 16, alpha 32, dropout 0.05; 3 epochs; learning rate $2e-4$; effective batch size 16 on 2 GPUs) on `ajayk007/querysmith-spider-bird`, derived from the Spider and BIRD benchmarks.
- **Novel method, partial-residual quantization:** aggressive quantization confined to a subset of transformer blocks, with the quantization error corrected by a low-rank SVD residual term.
- **Five-stage CPU study:** (1) precision sweep FP32, FP16, INT8, INT4; (2) quantization strategies: selective (MLP-only), partial (last-k blocks), residual (INT8 + low-rank SVD correction), and combined partial-residual; (3) GGUF conversion and llama.cpp benchmarks at F16, Q8_0, Q4_0; (4) CPU thread sweep over 1, 2, 4, 6, 8, 12, 16 threads; (5) re-fine-tuning the optimized model.
- **Evaluation:** 200 held-out examples scored by exact-match SQL accuracy, with execution accuracy against live databases as a check; CPU benchmarks on a 14-core Intel Core i5 (13th Gen) with 16 GB RAM.
- **Results (accuracy / ms per token / size in MB):** FP32 baseline 0.713 / 148.0 / 14,890.5; FP16 0.709 / 121.4 / 7,445.0; INT8 0.682 / 89.1 / 3,721.8; naive INT4 0.636 / 71.1 / 1,859.9; partial-residual 0.707 / 79.3 / 3,511.2; partial-residual + llama.cpp Q4_0 0.707 / 27.5; final model after re-fine-tuning 0.743 / 22.3 / 3,511.2.
- **Key findings:** naive INT4 cuts size 87.5% but loses 7.7 accuracy points; partial-residual recovers 92.2% of that loss (within 0.006 of FP32); llama.cpp adds a further 65.3% latency reduction; 8 to 12 threads is optimal; the final model beats the FP32 baseline (0.743 vs 0.713, +4.2% relative) while running 6.6x faster and 4.2x smaller.
- **Limitations reported:** residual rank fixed at 8 (not swept) and a modest 200-example evaluation set.
- **Links:** Paper zenodo.org/records/23000490
Notebook kaggle.com/code/maroofiums/text-to-sql-phi4-mini-qlora
Model huggingface.co/maroof2424/Edge4-SQL-CPU-Q4_K_M

An Empirical Study of Adaptive Activation Attention (A3)

May 2026

Zenodo preprint v1.0, single author (DOI 10.5281/zenodo.20364414)

- **Problem:** studied an attention mechanism that dynamically modulates activation responses during representation learning, evaluated across multiple experimental settings for learning stability, convergence behavior, representation quality, and computational efficiency.
- **Findings:** improved feature selectivity and optimization dynamics at competitive computational cost; best result was +2.1 percentage points over the baseline, with 91.8% validation accuracy, though gains over standard softmax attention were not consistent, particularly on short sequences.
- **Analysis:** examined architectural trade-offs and attention behavior, including gate collapse, gradient instability, and short sequence lengths masking any benefit.
- **Follow-up (A3-v2):** added a gate regularizer, long-context benchmarking with MQAR, and multi-seed bootstrap significance testing; runs on a free Colab T4 GPU for reproducibility.
- **Links:** Paper zenodo.org/records/20364414 | Code github.com/maroofiums/A3-Attention
Notebook kaggle.com/code/maroofiums/a-attention

Adaptive Retrieval Policies for RAG Systems

In progress

Manuscript (LaTeX)

- Research question: can a lightweight controller learn a retrieval policy that balances answer quality, faithfulness, and retrieval cost?
- Manuscript drafted with introduction through experimental protocol; baseline experiments (No-RAG and Standard RAG) are the next step.

PROJECTS

CognifyAI – AI Code Optimiser Platform

FastAPI, PostgreSQL, LangChain, React, TypeScript, Monaco Editor, Docker github.com/marroofums/CognifyAI

- Built a full production-ready AI platform that analyzes code and suggests optimizations, owning the project end to end: frontend, backend, database, LLM orchestration, and containerization.
- Results: 1.8 s response time, 42 automated tests, about 1,900 lines of code, 6-stage pipeline across 5 core technologies (FastAPI, PostgreSQL, LangChain, React/TypeScript, Docker).
- Designed a six-stage sequential analysis pipeline orchestrated with LangChain; each stage feeds the next.
- Implemented real-time NDJSON streaming so results appear in the UI as each stage completes.
- Frontend in React and TypeScript with the Monaco Editor (the VS Code editor engine) embedded.
- Backend in FastAPI with PostgreSQL; the whole stack runs in Docker.
- Debugged and resolved Docker DNS and local database connection issues in the containerized setup.
- Migrated the LLM provider from OpenAI to Mistral (`mistral-small-latest`).

LayaFlow – Decision and Workflow Automation System

Python, FastAPI, Pydantic, SQLAlchemy, PostgreSQL, Pytest, Docker github.com/marroofums/LayaFlow

- Built around Laya, a structured probabilistic decision model that turns an incoming request into 3 outputs: a category, a priority, and an escalation probability.
- Design goal: keep decisions explainable and cheap by using a small specialized model for the decision and a deterministic policy engine for the action, instead of calling a generative LLM for every request.
- Results: 94.2% Laya accuracy on a 1,200-sample dataset, validated with 100 evaluation tests.
- 4-stage pipeline: Laya model, decision normalizer (cleans and validates model output), policy engine (applies business rules), workflow engine (executes one of 3 actions: assign, escalate, or queue).

Edge4-SQL – Deployable CPU Text-to-SQL Model

llama.cpp, GGUF, Hugging Face [Hugging Face](https://huggingface.co/marroofums/Edge4-SQL)

- Published the CPU-optimized Q4_K_M GGUF build of the QuerySmith text-to-SQL model on Hugging Face for local CPU inference.
- Benchmark scope: 4 precisions, 4 quantization strategies, 5 stages, plus a CPU thread-count sweep, all on CPU only.
- Versus the FP32 baseline in the paper: about 85% lower latency per token (148 to 22.3 ms), about 76% lower memory (14,890 to 3,511 MB), and exact-match accuracy up from 0.713 to 0.743.

Math2Sound – Math Function to Audio Converter

Python, NumPy, FFT, WAV github.com/marroofums/Math2Sound

- Interactive Python app converting math functions ($\sin(x)$, x^2 , $\sin(x)+\cos(2x)$) into audio via selectable mappings.
- Covers sampling, frequency, amplitude, phase, PCM and WAV encoding, with FFT and spectrogram analysis.
- $\sin(x)+\cos(2x)$ has 2 frequency components, so the FFT output can be checked directly against the equation.

AI Research Assistant (RAG System)

Python, LangChain, RAG github.com/marroofums

- Retrieval-augmented assistant for querying and summarizing research documents.
- Results: indexed 25 research documents (500-token chunks) and reached 88% retrieval accuracy.

Movie Recommender System (Siamese Transformer)

PyTorch, Transformers, Deep Metric Learning github.com/marroofums/Movie-Recommender

- Built a content-based movie recommender that learns similarity between movies directly, instead of relying on user rating history, so it can recommend items with no interaction data (cold-start).
- Architecture: a Siamese network where two weight-shared Transformer encoders each turn a movie's text into an embedding vector; the distance between the two vectors is the model's similarity score.
- Training: pairs of movies are labeled similar or dissimilar and the shared encoder is optimized so similar movies sit close together in embedding space and dissimilar ones sit far apart.
- Inference: embed the query movie once, then rank the catalog by similarity to return the top-N recommendations.
- Results: trained on 45,466 movies; Recall@10 of 0.67.

Deep Learning Projects

PyTorch, OpenCV, Scikit-learn

github.com/maroofiums

- **Environmental Sound Classification (ESC-50):** PyTorch CNN on log-Mel spectrograms; 91.4% accuracy (ESC-50 has 2,000 five-second clips across 50 classes). [GitHub]
- **News Classification:** PyTorch LSTM over 4 categories, 93.1% accuracy, served through a REST API. [GitHub]
- **Emotion Classification:** BiLSTM text classifier over 6 emotion classes, 89.6% accuracy. [GitHub]
- **Face Mask Detection:** MobileNetV2 transfer learning with OpenCV for real-time detection; 96.2% accuracy. [GitHub]
- **Others:** BERT news classifier, HAR CNN-LSTM, neural network from scratch, churn and loan prediction, face detection API.

EDUCATION

Diploma of Associate Engineer (DAE), Software Technology

Aug 2024 – Sep 2027

MITI Karachi

Karachi, Pakistan

- Status: currently in 3rd year; expected completion September 2027. Strong academic performer who regularly takes part in exhibitions.
- **Web Development (HTML, CSS, PHP, CodeIgniter, React, SEO):** building responsive websites, server-side PHP applications with the CodeIgniter MVC framework, component-based interfaces with React, and search engine optimization (SEO). *Applied in:* the React and TypeScript frontend of CognifyAI.
- **Programming and Software Engineering (Java, C#, .NET, OOP, Android):** object-oriented design (encapsulation, inheritance, polymorphism, abstraction), application development in Java and C#/.NET, and building Android applications. *Applied in:* my LCI internship, where the AI agent integrates with an internal .NET system.
- **Databases and Enterprise Systems (SQL, Oracle ERP):** relational database design, SQL querying, and business process workflows in Oracle ERP. *Applied in:* PostgreSQL backends in CognifyAI and LayaFlow, and my text-to-SQL research (QuerySmith).
- **Professional Skills (Business Communication):** professional writing and presentation, practiced at exhibitions and when presenting results to senior developers.
- Alongside the diploma, self-taught ML and AI backend engineering: 3 internships, 2 published research papers, a Kaggle rank of 335, and 30+ public repositories.

ACHIEVEMENTS

- **Kaggle Rank 335:** ranked 335 on Kaggle, within the top 400 (kaggle.com/maroofiums).
- **Published researcher:** 2 research papers published on Zenodo (QuerySmith text-to-SQL; Adaptive Activation Attention), with a third paper on adaptive RAG retrieval policies in progress.
- **CPU efficiency result:** cut text-to-SQL inference latency by about 85% (148 to 22.3 ms/token) and memory by about 76% (14,890 to 3,511 MB) while improving exact-match accuracy from 0.713 to 0.743.
- **Open-source model release:** published the CPU-optimized Edge4-SQL model on Hugging Face (Hugging Face link).
- **Production delivery:** shipped CognifyAI as a full-stack, Dockerized AI platform with a 6-stage pipeline, 42 automated tests, and 1.8 s response time.
- **Industry experience:** completed 2 production internships (DeveloperHub Corporation, Elevvo Pathway) and started a software engineering internship at Lucky Core Industries in September 2026.
- **Model performance:** 94.2% accuracy on LayaFlow's decision model, 96.2% on face mask detection, and 93.1% on news classification.
- **Open source:** 30+ public ML/AI repositories on GitHub (github.com/maroofiums).
- **Exhibitions:** regular participant in exhibitions, presenting technical projects.