Yunbeom Choe

Machine Learning Engineer | Production ML Systems | Deep Learning

Liverpool, UK (Open to Relocate) | +44 7826 338205 | yunbeom.choe.dev@gmail.com

choeyunbeom.github.io (technical blog) | LinkedIn | GitHub

PROFESSIONAL SUMMARY

Machine Learning Engineer building production ML systems across LLM, computer vision, and reinforcement learning domains (MSc Data Science & AI, University of Liverpool, graduating Sep 2026). Diagnosed a 28pp LoRA fine-tuning regression to training data contamination, optimised retrieval pipelines (MRR 0.51→0.82), and deployed PatchCore anomaly detection at 47ms latency via OpenVINO — backed by 208 automated tests and Dockerised microservices. UK right to work: full-time from Oct 2026 (Student visa), then Graduate visa from Dec 2026 — no sponsorship required for 2 years; Skilled Worker sponsorship-eligible at new entrant rate thereafter.

EDUCATION

University of Liverpool *Sep 2025 – Sep 2026*

MSc Data Science & Artificial Intelligence

- Modules: Applied AI, Machine Learning, Computational Intelligence, Research Methods, Databases, Mathematics & Statistics.

Chonnam National University *Mar 2017 – Feb 2024*

BEng Naval Architecture & Ocean Engineering (incl. 21 months national military service)

PROJECTS

arXiv Academic Paper Q&A System (RAG) | Personal Project *Feb 2026*

github.com/choeyunbeom/arxiv_rag_system

- **Problem:** Baseline vector search over 153 academic papers returned relevant results only 60% of the time, making the Q&A system unreliable for research workflows; needed systematic retrieval optimisation and evaluation methodology to diagnose component-level failures.
- **Solution:** Built a systematic retrieval optimisation pipeline: chunk size tuning, BM25 hybrid search with Reciprocal Rank Fusion, and cross-encoder reranking. Conducted LoRA fine-tuning on Qwen3 4B (0.81% trainable params) and ran 3-way evaluation against baselines on a held-out 50-query set generated from a separate paper subset to prevent leakage. Deployed as Dockerised microservices (FastAPI + Streamlit) with 208 automated tests (unit + integration) via GitHub Actions CI.
- **Result:** MRR 0.51 → 0.82, Hit Rate@5 60% → 100% (50/50 queries). Fine-tuning evaluation revealed a 28pp performance drop traced to training data contamination (Thinking-mode artefacts in the QA dataset) — a failure analysis that now serves as my reusable evaluation methodology for LLM system reliability.

Reply Mirror: Multi-Agent Fraud Detection | Reply Challenge 2026 *Apr 2026*

github.com/choeyunbeom/reply_ai_chal

- **Problem:** Financial fraud patterns evolve continuously — shifting merchants, time windows, geographies and amounts — causing static detection models to degrade across successive data releases.
- **Solution:** In a 3-person team during a 6-hour sprint, designed a 6-agent system: Orchestrator routes transactions through Isolation Forest, Context Agent, and LLM Investigator (Popperian falsification + Bayesian updates), with Critic and Memory agents. Owned Role A (Pipeline & Orchestration Owner): pipeline integration across 6 agents, cost tracking ($160 LLM budget), and submission validation.
- **Result:** Top 7% (137th / 1,971 teams). System adapted across 5 levels of concept drift without retraining on labelled data, demonstrating multi-agent orchestration, cost-constrained LLM deployment, and drift-resilient anomaly detection.

FinScope: Multi-Agent Financial Filing Analyst | Personal Project *Mar 2026*

github.com/choeyunbeom/finscope

- **Problem:** Equity analysts spend hours cross-referencing risk, growth, and competitive disclosures across 10-K filings — and single-LLM RAG approaches frequently hallucinate citations on regulated financial documents.
- **Solution:** Architected a LangGraph multi-agent pipeline (Retriever → Analyser → Critic → Query Rewriter) over SEC EDGAR and Companies House APIs. Critic agent triggers feedback-driven query reformulation when >30% of claims lack

citations (semantic equivalence check). Achieved 1.6s parallel vs 4.9s sequential analyser latency (3.1x speedup, measured) via asyncio.gather for parallel LLM calls (Groq/Llama-3.3-70b). Designed agent state with TypedDict, used checkpointing for graph recovery, and integrated Langfuse for end-to-end LLM call tracing. Validated with 35 tests (24 unit + 11 integration) covering full graph state transitions.

- **Result:** On a 9-case synthetic evaluation (clean / hallucinated / borderline), 70B Critic achieved 100% sensitivity and 83% accuracy, catching 3/3 borderline cases (~30% altered numeric claims) while 8B caught 0/3 — confirming model size as a critical factor in boundary-case hallucination detection. Both models showed 67% specificity, with paraphrased clean content occasionally misclassified as uncited — an actionable failure mode for next iteration (threshold tuning, claim-level evaluation). Full audit trail of all evaluation runs in Langfuse.

DefectVision: Manufacturing Defect Detection | Personal Project *Mar 2026*
github.com/choeyunbeom/defectvision

- **Problem:** Supervised defect detectors require hundreds of labelled defect images per type and cannot detect novel defects — impractical for manufacturing lines where defects are rare and unpredictable.
- **Solution:** Built a real-time anomaly detection system using PatchCore (unsupervised, trained on normal images only). Evaluated on MVTec AD and the harder MVTec AD 2; tested augmentation strategies that degraded performance, identifying memory-bank pollution as root cause. Optimised inference with OpenVINO and deployed as a FastAPI + Streamlit pipeline with threaded webcam capture and on-site threshold calibration.
- **Result:** Demonstrated an 8–20pp performance drop from the saturated MVTec AD benchmark to the realistic MVTec AD 2, and identified data coverage — not augmentation — as the correct mitigation for memory-bank methods. Deployed as a production-ready pipeline with 47ms model inference (~6 fps end-to-end pipeline).

Autonomous Racing Agent | IBM AI Racing League *Jan 2026*
github.com/choeyunbeom/ibm_ai_race

- **Problem:** RL agents failed to complete a 3,600m track — exploiting reward functions by "parking" at high-reward positions rather than racing.
- **Solution:** Trained PPO and SAC agents over 9.7M steps with custom reward engineering (distance-scaled crash penalties, low-speed termination). Conducted telemetry-based failure analysis to diagnose reward exploitation and a 2,400m completion plateau — extending the same failure-mode-driven debugging methodology applied across my arXiv RAG (28pp regression) and DefectVision (memory-bank pollution) projects.
- **Result:** 37 autonomous completions, best lap 1:48. Qualified for next round of IBM AI Racing League. Documented systematic debugging process in a technical blog post covering reward shaping iterations.

TECHNICAL SKILLS

- **ML & Deep Learning:** PyTorch, scikit-learn, LoRA/QLoRA Fine-Tuning, Hugging Face Transformers, Anomaly Detection (PatchCore, Isolation Forest), Reinforcement Learning (PPO, SAC), Stable Baselines3, Anomalib
- **LLM & AI Systems:** RAG Systems (Hybrid Search, BM25, RRF, Reranking), LangChain/LangGraph, Multi-Agent Architectures, Hallucination Detection, Ollama, Groq API, Langfuse
- **Data & Infrastructure:** Python (NumPy, Pandas, Matplotlib), SQL, ChromaDB, Vector Search
- **MLOps & Deployment:** Docker, FastAPI, Streamlit, OpenVINO, GitHub Actions CI/CD, uv, pytest, Azure (AI-900 certified)

CERTIFICATIONS

Microsoft Azure AI Fundamentals (AI-900) *Nov 2024*

OTHER EXPERIENCE

Republic of Korea Army — Ammunition Depot Handler *Feb 2018 – Oct 2019*

- Maintained safety-critical ammunition inventory in a zero-error, audit-driven environment — discipline now applied to testing and operational reliability in ML systems.