College of Information, Data, and Society
Department of Applied Data Science

Applied Data Science Expo Day

Showcasing student-led innovations

2026

Applied Data Science Department

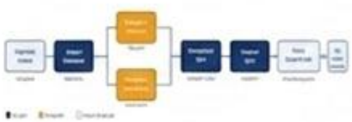
NexusTrace: Multi-Agent Collaboration System for Kubernetes Error Log Analysis

Project Advisor: Dr. Simon Shim & Dr. Lee C. Chang

Alarcon, Najai
Jaculina, Chalssa
Katta, Mrunali
Singh, Devesh
Thiagarajan, Aksah

Introduction

Kubernetes envirenmenta generate massive volumas of logs and orev avents requiring deep expertise tn diognase. Single LLM systems frequendy hafsemets and produce uncole remodiation. NaxusTrace coordinates four acceialkied LLM agents using a Blackboard shared memory architecture two pwerieil RCA agents. A Recanciliation agent, and a Validation agent covering 19 Kubemetes farure categories with a striot human in-the-kop approud gata.



- 9,500 balanced synthetic Kubernetes incident records across 19 failure categories.
- Models: Qwen 3.5.5B, DeepSeek-R1 8B, Devenal-Small-2 21B, Llama 3.2 3B
- Deployed on AWS EXS via Docker containers with LoRA fine-tuned adapters.

Methodology

Multi-Agent Orchestration Pipeline

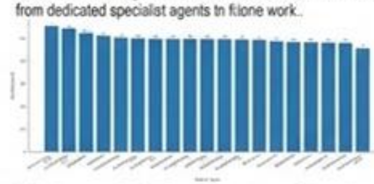
NexusTrace uses four LLM agents. has perallol RCAs, a reconoliation step for remadiation, and a validaior for rollback/safely all gated by human approval. Adapters were fine-funed and Aun on AWS EKS.

- Data: 9.500 cyntnetic Kits incidents → transformed into 6 sele-specific SFT corpora
- Training: GLaRA 4-b4 few-turing with LoRA adapters (r=18) on Google Colsb A10011109
- Deployment: Docker images poshnot to Decker Hub → dsployed on AWS EKS (oovav trace namespace)

Analysis and Results

NexusTrace was evalusied across two profiles, end-to-and orcinustration on the depleyed EKS etwiss using Ilama3.3 5t, and standsione SFT evaluation of the offlinani/ained Qwen2.5-TB-hstruct soRA adapter on a 406-example held-out set.

The SFT evaluation reported scenarie-mahh scervasy of 0.715, section-evoremeos of 8.5T3, and BSRTScure F1 of 6.841. Five scenanis reached perfect accuricy including OOMkiled, QuotaExcoded, LivenessProbeFatied, PVCNorFeund, and ConfigclupError. Manker svonarica auch as FailedSchorviing and ReceRelasionMismatah wit benefit from dedicated specialist agents tn filone work.



All five themed erchoilzation paylaats: storage fallures, image-soll arrors, scheduling errors, runoms oreshes, and RBAC dertals which produco ovitdanco-wwece diognoses and slmnicoesi remciliation plans with warm inference latency avereging 10–25 seconds per agent call.

Evaluation Notrics

Schema valldation baased on all 400 SFT evaluation exemples with zaro mismatch errors. The Pydantic-validated RCA structure coveting diagnosis, likely causes, remadiation dalas, rolfeacs plan, and verfication commatios round-mpped through MongdBB and twe frenlond renderer without fatures.

The end-tn-end ingeat srpeline ran continuoudy wihoui message ross across deliberate pod restarts, confvming ts-iwasl-once Kafka delivery. All orchestration evaluation sutes passed with condistent behavior in both stab-wsale and kca model tasling.

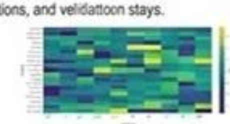
Per-scenario accuracy ranged from 1.00 on OCAfiOled, Guesokisiscdon, and LivenssProbeFalled to 0.24-0.47 an FailedScheduling and NateSelesterMamatsh. Falsves with stereetyped event text were easiest to diagrawe while scheduling and registry-outh sconorics were hardest. Dukve specialst agents for nstworking, storage, and securlty will address these gaps.

0.972	0.844	0.715	10-25s
Seeclon Sorse	BEBTSrors F1	Socenelic Emcik	Warn Lamey

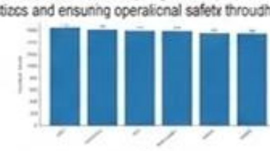
9,500	19	400

The Blackboard shared-mamory archksatore ensured full mungdi0ls aoross every reasoning stage. Each inokiaed in MongeGB carries Kafta offset metadass and an ingested at timestamp, as any RCA output can be traced back to the exact-alert that triggered it.

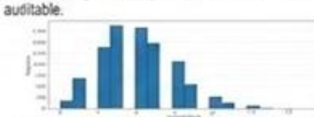
Two RCA agents analyzed each heident independently using different models and reasoning strategies. The Reconciliation agent enrirated disagreements and produced a unilled remniliation plan wiln kubseti conmends, withack instructions, and velidatoon stays.



The human-in-the-loop approval gate prevented any infrastructure-changing commtnd from axeeuting wlthaut explioct operator review, aligning with AI genemafize best proctizes and ensuning operalional safetx throudhout.

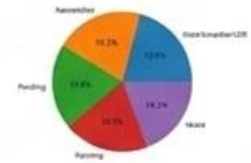


NexusTrace demonstratsd that collaborative LLM reasoning on raul Rubernetes hifastrusture mataralty improves incident diagroccc compared to single-model approachos while manitaiving strict operativnis safety through human averiiple.
The GKCps-managed deployment surfiore ansured full reproducibility where every ciester changa is materialized frum Gil by Flax, an the estio stavs from Kafka through Citama through the analyser is version-controlled and auditable.



Kastart poants peaked beavean 2–5, consistent with rent Kubernetes CrachLoopSacAOR bchaiwv. The timeline confirms stable data generation with 1,400+ invidents daily through January 6 before generation completed.

Resource usage was broadly disiributed across of CPU and memory ranges, canfinimg dinorse operational coverage across fafore scenatics.



Summary/Conclusions

NaxusTrace shows has collaborative multi-agent LLM reasoning inroroves Kubernetes ineident degnesis compared with single-model workfloms. By comirtiing paratal RCA, reconciliation, and validation, the system prodwoes clearer and safer remodiation guidanes with reilbavk rinusos.
The platform is deployment ready through Docker and AWS EKS, and supports beth lightweight and larger models. Overall, this work shows a prastical, troztiverlhy framework for AI-essxord-cloud approvens.

Key References

[1] Burns, B., Bomt, B., Oppenheimer, D., Brewer, E., & Wilres, J. (2016). Borg, Omega, and Kabernetes.

[6] Bums, B., & Beda, J. (2019). *Kubornetes: Up and Running* (2nd ed.).

[3] Bal, J., et al. (2024). *Qwen2.5 Technical Report.* arl0v:2412.15115.

[4] Tian, R., et al. (2023). *DespSeek-R1 Technical Report.*

[8] Hu, E. J., et al. (2021). *LcRA: Low-Rank Adaptation of Large Language Models.* arXiv:2196.25665.

[8] Dattmers, T., at al. (2023); *QLoRA: Efficient Finetuning of Quantized LLMs:* arfiv-2305.14314.

Acknowledgements

We thank Dr. Simon Shim & Dr. Lee C. Chang for all the guidance and support. We also thank the Applied Data Science Department at San Jasa Stats University and our team members for collaboration throughout this project.

Applied Data Science Department

Personalized Education with Semantic Memory & RAG

Project Advisors: Dr. Simon Shim, Dr. Lee C. Chang

Basanth Periyapatna Roopa Kumar
Manav Rajesh Anandani
Nandhakumar Apparsamy
Nischitha Nagendran
Nitya Rondla

Introduction

The Personalized Education Agent addresses the "personalization gap" in e-learning by converting static academic materials into an adaptive, document-grounded AI tutor. Standard LMS platforms and general AI assistants often lack access to a learner's specific course materials, causing context-blind responses and hallucinated explanations.
Our system combines Retrieval-Augmented Generation, ChromaDB-based semantic memory, LlamaParse document ingestion, and an adaptive Root-Cause Analysis engine. The final prototype generates personalized learning paths, supports grounded Socratic tutoring, visualizes mastery through a ReactFlow concept graph, and inserts remedial milestones when prerequisite gaps are detected. Final evaluation showed hallucinations reduced from **15.4% to <2%**, *a* **0.94 RAGAS faithfulness score**, **680ms average time-to-first-token**, *and* **94% RCA diagnostic accuracy**.

Methodology

The learner uploads course material and a learning goal, which initiates the pipeline. Documents are parsed using LlamaParse to preserve structure such as tables and formulas. The content is split using semantic chunking, and each chunk is converted into embeddings and stored in ChromaDB.

At query time, relevant chunks are retrieved using similarity search (Top-K) and passed into a RAG pipeline to generate grounded responses. The system uses low-latency LLM inference to ensure real-time interaction.

A Root-Cause Analysis (RCA) engine analyzes quiz performance to identify missing prerequisite concepts. Based on this, the system dynamically updates the learning path by inserting refresher milestones.

Outcome: Static PDFs are transformed into an adaptive, personalized, and document-grounded learning experience.

Analysis and Results

Introduction: Context Blindness in AI Tutors
Your Most e-learning platforms and general AI tutors do not understand a student's exact course materials, professor-specific notation, or current knowledge gaps. As a result, students often receive generic answers that may not match their uploaded textbooks, lecture notes, or class expectations.

This creates three core problems:

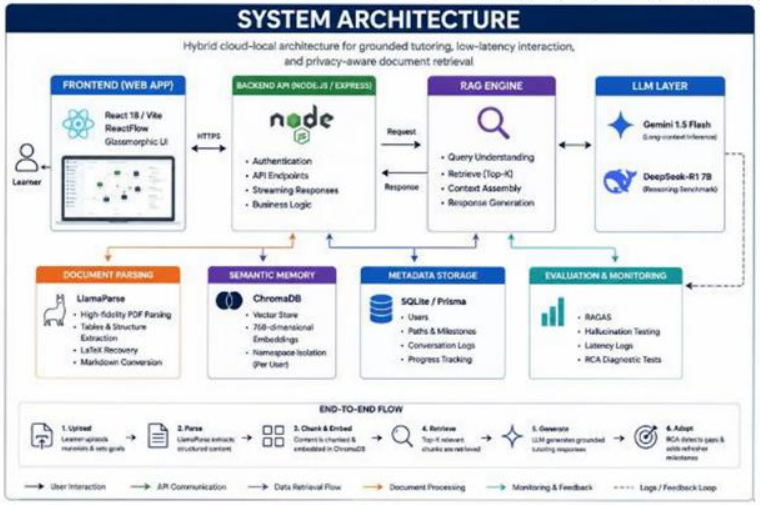
One-size-fits-all learning: static learning paths ignore individual mastery.

Hallucination risk: AI may generate unsupported or incorrect explanations.

Cold-start burden: students must manually turn dense PDFs into study plans.

Goal: Build an AI tutor that reads, remembers, retrieves, teaches, and adapts using the learner's own curriculum.
paragraph text

Technology Stack Section
The system uses a hybrid cloud-local architecture designed for grounded tutoring, low-latency interaction, and privacy-aware document retrieval.



Intelligent Features Section

Ingestion Hub: Upload materials + goal.
AI Tutor: Grounded lessons and Q&A.
Concept Map: Mastery, gaps, dependencies.
RCA Panel: Finds prerequisite gaps.
Linear Syllabus: Personalized milestones.
Quiz: Captures mastery signals.
Flow: Upload → Path → Learn → Quiz → Gap → Adapt.

Summary/Conclusions

The Personalized Education Agent shows that AI tutoring can move beyond generic Q&A into grounded, adaptive learning support.
By combining RAG, semantic memory, LlamaParse ingestion, ReactFlow mastery visualization, and RCA-based remediation, the system reduces hallucinations while improving explainability.
Key takeaway: A student's private curriculum can become an interactive learning environment instead of a static collection of PDFs.

Key References

[1] Bloom, B. S. (1956). Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain. Longmans, Green.

[2] Google Research. (2025). Learn Your Way: Reimagining textbooks with generative AI. Google Research Blog.

[3] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459–9474.

[4] Li, Z., & Wang, X. (2025). Retrieval-augmented generation for educational application: A systematic survey. Computers and Education: Artificial Intelligence, 8, Article 100392.

Acknowledgements

We thank Dr. Simon Shim and Dr. Lee C. Chang for their guidance, technical feedback, and support throughout the DATA 298A/B project lifecycle. We also thank the Department of Applied Data Science at San José State University for providing the academic structure and resources needed to complete this capstone project.

Graph-based Few-shot Threat Detection Using RAG Models

Applied Data Science Department

Project Advisor: Dr. Simon Shim
SJSU MSADI 298B | May 2026

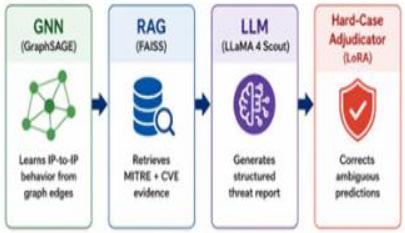
Smit Ardeshana
Devarsh Patel
Dharmitkumar Patel
Lovely Priya
Mansi Tanna

Introduction

Research Problem
- SOC analysts face high-volume network flow telemetry with limited labeled attacks
- Attack classes are highly imbalanced and rare
- Malicious behavior appears as **relationships between hosts**, not isolated rows
- Existing models lack **explainability and structured reasoning**

Research Gap
- Flat ML models fail to capture relational attack patterns
- LLM-only approaches lack structured classification reliability
- Missing integration of **graph learning + threat intelligence + reasoning**

Proposed Solution

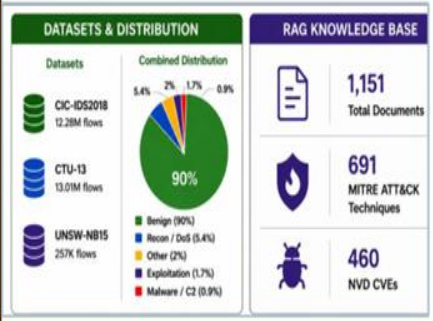


Methodology

End to End Pipeline:
Flow → GNN → RAG → LLM → Adjudicator → Report

Key Idea
- GNN → structural behavior
- RAG → CVE + MITRE context
- LLM → explanation
- Adjudicator → fixes hard cases

DATASETS & KNOWLEDGE BASE

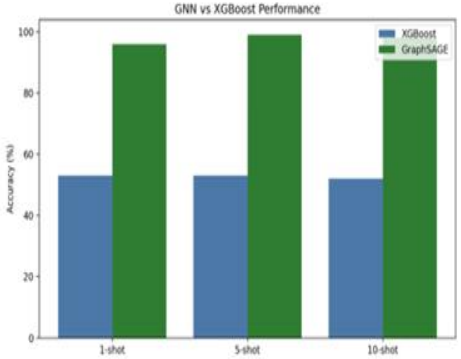


Analysis and Results

Key Results

3.1 GNN Performance vs Baseline

GraphSAGE significantly outperforms XGBoost baseline across few-shot settings.

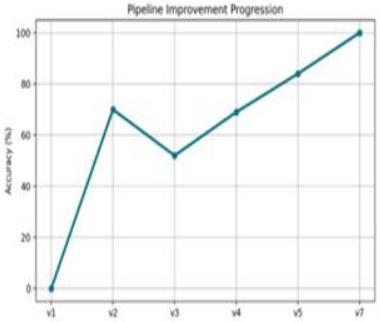


3.2 LLM + RAG Comparison

Model	Quality	JSON	Speed
LLaMA 4 Scout	87.5	100.0	0.73
Qwen 32B	84.8	100.0	4.12
Kimi K2	82.6	100.0	1.34
LLaMA 3.1	76.3	100.0	0.53
LLaMA 3.3	100.0	2.5	31.0

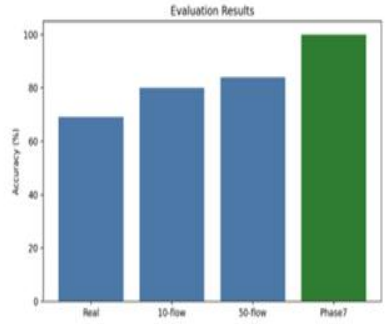
RAG Impact:
Without RAG → **17% accuracy**
With RAG → **84% accuracy**
(+67 percentage points improvement)

3.3 Pipeline Improvement Progression



Accuracy improves from 0% to 84% after rule-based refinements and RAG integration.
Phase 7 specifically targets hard cases using adjudication.

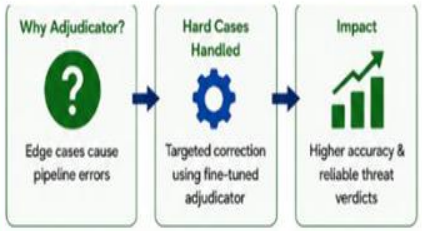
3.4 Evaluation Results



Real dataset: 69.2%
10-flow challenge: 80.0%
50-flow challenge: 84.0%
Phase 7: 100% (hard cases)

Contribution & Conclusion

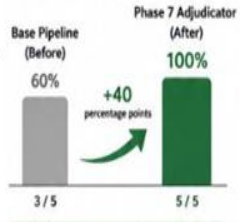
4.1 Phase 7 Adjudicator



Hard Cases Fixed
- Slow C2 (HTTPS 443)
- DNS tunneling
- No-GNN C2 (IP mismatch)
- PPS artifacts (short flows)

5. CONFIGURATION
- Base: LLaMA 3.1 8B
- Method: LoRA (4-bit NF4)
- Trainable: **0.52%**
- Data: **150 samples**
- Training: **3 epochs**

Impact (Hard Case Accuracy)



100% accuracy on all hard cases

Evaluation & Conclusion

6. EVALUATION
- Real dataset: **69.2%**
- 10-flow: **80%**
- 50-flow: **84%**
- Phase 7 (hard cases): **100%**

7. CONCLUSIONS
- GNN achieves **F1 = 0.989**
- RAG improves **17% → 84%**
- Adjudicator fixes edge cases (**60% to 100%**)
- Fully integrated explainable system

Limitations & Future Work

8. LIMITATIONS
- High latency (~89s/flow)
- Limited training data
- Not real-time yet

9. FUTURE WORK
- Optimize latency
- Expand dataset
- Real-time deployment
- SOC/SIEM integration

References

- Hamilton et al., GraphSAGE
- Lewis et al., Retrieval-Augmented Generation
- MITRE ATT&CK Framework
- Hu et al., LoRA

LEXIS - A LEGAL ANALYSIS SYSTEM USING AI

Applied Data Science Department

Project Advisor: Dr. Simon Shim, Dr. Lee Chang

Team 4

Adhyaru, Kushal
Chao, Brendan
Kothuru, Mayuka
Ong, Albert
Shah, Prem

Introduction

Legal contract review is manual, elow, and costly. Lawyers spend hours identifying obligations, flagging risks, and verifying compliance across dense legal documents. Standard LLMs fail to reliably distinguish between obligations, prohibitions, and permissions without domain-specific fine-tuning. Commercial platforms are proprietary and opaque.

LEXIS addresses this gap with an open, end-to-end legal intelligence platform. For extracting structured obligations from contracts, assessing risk, notifying similar provisions, and integrating directly into legal workflows through Google Calendar and Gmail.

Built on the CUAD dataset with 510 contracts and 41 clause types, LEXIS fine-tunes two open-weight LLMs using QLoRA and uses a six-step analysis pipeline that transforms raw contract text into workflows, querying obligation records.

Methodology

Six-Step Pipeline

Step 1: Span Extraction using ModernBERT with LegalBERT fallback.

Step 2: Deontic Classification into OBLIGATION, PROHIBITION, or PERMISSION

Step 3. Structured JSON Correction extracting action, owner, deadline, condition, consequence, and risk level

Step 4: Risk Assessment scoring HIGH, MEDIUM, or LOW

Step 5: Semantic Search using Jaccard similarity against stored clauses

Step 6. Calendar Integration mapping deadlines to compliance calendar

All services include fallback and regex fallbacks ensuring results are returned even without GPU access.

Analysis and Results

ML Pipeline

Model Performances

- Phi-4 and Qwen2.5-7B achieved the strongest post-fine-tuning on CUAD
- QLoRA fine-tuning on CUAD Deontic Accuracy and +28.9% in RGUGE-L
- All models reached near perfect Field Completion (87.8–100%), confirming consistent structured JSON output.

Baseline vs Fine Tune Model

Key Features

- Six-step automated obligation extraction pipeline processing PDF, DOCX, RTF, and TVT contracts in under one second with 84% deonts classification accuracy
- Interactive force-directed obligation network graph and compliance standard visualizing relationships and deadlines color coded by risk level across the full contract portfolio.
- 175 passing tests covering unit, integration, and ten real-world contract scenarios including NDA, SaaS, Employment, GDPR, and Mergers and Acquisitions
- Acquisitions

Summary/Conclusions

LEXIS proves that open-weight fine-tuned LLMs can deliver accurate, structured legal analysis without proprietary systems. QLoRA fine-tuning produced a 23-point deontic accuracy improvement and 100% JSON validity across all models. The six step pipeline with graceful degradation ensures reliable performance in both development and production environments.

LEXIS work includes multi-language support, human-in-the-loop feedback, knowledge distillation for smaller deployment, and real-time contract monitoring via webhooks.

Key References

[1] Hendrycks et al. CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review. 2021.

[2] Tuggener et al. LEDGAR: A Large-Scale Multi-Label Corpus for Legal Provisions. LREC 2020.

[3] Chalkidis et al. LexGLUE: A Benchmark for Legal Language Understanding. ACL 2022.

[4] Chalkidis et al. LEGAL-BERT: The Muppets Straight Out of Law School. 2020.

[5] Siino et al. Exploring LLMs Applications in Law. IEEE Access 2020.

[6] Shu et al. LawLLM. Law Large Language Resist for the US Legal System, CIKM 2021.

Acknowledgements

We are grateful to the Applied Data Science Department for the resources and academic support that made the development of the LEXIS project possible.

GitHub: github.com/mayuka-reddy/298B

Real-Time Sentiment Analysis & Customer Support Automation

Applied Data Science
San José State University
DATA 298B · Team 5

Project Advisors: Dr. Simon Shim · Dr. Lee C. Chang

Anshu Reddy Dhamana
Sritharedy Devireddy
Akhil Kumar Gudapuri
Vinuthna Papana
Chandana Sreya Reddy

Introduction

Business Problem

E-commerce platforms handle thousands of support requests daily — refunds, order tracking, payment failures, complaints. Traditional rule-based chatbots and manual ticket handling cannot understand customer intent, emotional tone, or urgency, causing delayed responses, poor prioritization, and inconsistent experiences.

Project Goals

Build an end-to-end AI customer support platform that performs intent classification, sentiment analysis, urgency prediction, named entity recognition, out-of-scope detection, and contextual response generation in real time — and automatically escalates critical complaints.

Project Scope

The system unifies six concurrent NLP tasks — intent, sentiment, urgency, NER, OOS detection, and response generation — in a single orchestration pipeline with sub-2-second end-to-end latency. The modular architecture supports independent task fine-tuning, cloud-native deployment with structured logging, and a SHOP9 e-commerce demo with full order-management integration validated across DATA 298A and 298B milestones.

Expected Business Impact

The platform delivers faster response times by automating triage of high-volume issues and routing critical complaints to the right human agent before churn becomes a risk. The same modular design transfers cleanly across e-commerce, SaaS support, banking, telecom, healthcare, and digital service workflows.

Project Outcomes (Targets)

- **Sentiment Accuracy:** ≥ 80% → **82.7% achieved**
- **Intent Recognition:** ≥ 80% on 27 classes → **84.3% achieved**
- **Urgency F1:** ≥ 0.75 → **0.786 achieved**
- **End-to-End Latency:** < 2 seconds → **< 1.5s avg**
- **Production-Ready Stack:** React + FastAPI + Cloud → **Docker + Cloud Run**
- **Cloud Logging & Monitoring:** BigQuery analytics → **Operational**

Stakeholder Need

Customer support teams need automated triage that identifies frustrated and urgent customers in real time, reducing manual review and routing time while preserving response quality on critical complaints.

Methodology

5-LLM Multi-Model Pipeline

- **Qwen2.5-3B** — Intent classification (27 classes)
- **Phi-3.5-mini** — Sentiment analysis
- **Gemma-3-1B** — Urgency prediction (3 levels)
- **GLiNER-large** — Zero-shot named entity recognition
- **BGE Embeddings** — Out-of-scope detection
- **Mistral-7B** — Contextual response generation

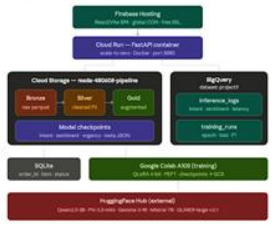
Training & Optimization

QLoRA fine-tuning on Google Colab GPU. Weak-supervision labeling for urgency. Synthetic e-commerce augmentation: word drop, swap, filler insert, truncate, synonym, informal-style rewrites.

Data Engineering Pipeline

Raw → Bronze: GCS parquet, run-ID versioned
Bronze → Silver: PII masking, dedup, normalize
Silver → Gold: Augment, balance, tokenize
Splits: Stratified 70 / 15 / 15, leakage-free

~26K Bitext intents • 650K Yelp reviews • 58K GoEmotions • 23K CLINC150 → 30,182 intent / 7,473 urgency Gold-layer training records.



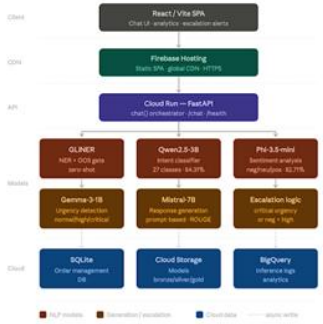
Analysis and Results

System Architecture

A modular orchestration platform routes each customer message through five specialized models in sequence, then assembles a context-aware response.

End-to-End Flow

1. React frontend — chat UI, analytics, order dashboard
2. FastAPI backend — orchestration & API routing
3. NLP pipeline — Qwen → Phi → Gemma → GLiNER → Mistral
4. SQLite — order lookup & retrieval
5. GCS + BigQuery — dataset storage & inference logs
6. Docker + Cloud Run — containerized deployment



Live Pipeline — Real Customer Queries

Customer: "Someone used my card without permission — this is fraud!"
Intent: complaint Sentiment: negative Urgency: CRITICAL
ESCALATED → "I sincerely apologize. This is being escalated to a senior agent immediately. Case ID generated."
Latency: 1.522 ms

Customer: "I've been waiting 2 weeks, this is completely unacceptable"
Intent: complaint Sentiment: negative Urgency: HIGH
ESCALATED → "I understand your frustration. I am escalating this to our priority resolution team right away."
Latency: 1.348 ms

Customer: "yo where my package at bro"
Intent: track_order Sentiment: neutral Urgency: low
AUTO-HANDLED → "I can help track your order right away! Could you share your order number?"
Latency: 1.391 ms

Escalation & Routing Logic

CRITICAL urgency triggers automatic escalation, generating a Case ID and notifying a senior agent immediately. HIGH urgency routes the query to the priority resolution queue with full conversation context preserved. NORMAL urgency is auto-handled by Mistral with order context retrieved from SQLite, freeing human agents for complex cases. Out-of-scope queries return a safe fallback message rather than allowing model hallucination, and any negative-sentiment complaint is automatically prepended with an empathetic apology template before the generated response, ensuring tone-appropriate communication regardless of which path the query takes through the pipeline.

Entity Extraction in Action

GLiNER pulls structured fields from messy customer text without requiring labeled training data, then feeds those entities back into Mistral as additional context — grounding generated responses in real order data instead of generic placeholders. The supported entity types cover the full vocabulary an e-commerce support workflow needs: order_id (e.g., ORD-12345, #4567, "order number 9876"), product_name ("wireless headphones", "Samsung TV"), tracking_number (1Z999AA10123456784, FedEx-style tracking IDs), date_reference ("March 15", "2 weeks ago", "yesterday"), payment_method ("credit card", "PayPal", "Apple Pay"), amount ($129.99, ₹2,500, "around fifty bucks"), and issue_type ("damaged", "wrong item", "missing").

Entity-Aware Response Pattern

GLiNER first extracts entities from the raw query text. If an order_id is present, the backend performs a SQLite lookup to retrieve order status, items, and history. Mistral then receives the original query plus extracted entities plus order context as a structured prompt, generating a response that references actual order details rather than generic fallback text. When a lookup fails because the order_id is malformed or not found, the system gracefully requests clarification rather than fabricating a confident-sounding response.

Final Evaluation — All Tasks

Each model evaluated on its target benchmark with stratified test splits. End-to-end latency measured on the deployed pipeline.

- **Intent Classification** — Qwen2.5-3B: 84.31% acc / 0.841 F1 ✓
- **Sentiment Analysis** — Phi-3.5-mini: 82.71% acc / 0.819 F1 ✓
- **Urgency Detection** — Gemma-3-1B: 79.41% acc / 0.786 F1 ✓
- **NER Extraction** — GLiNER-large: > 93% (zero-shot) ✓
- **OOS Detection** — BGE Embeddings: > 95% recall ✓
- **Response Generation** — Mistral-7B: ROUGE-1 0.42 ROUGE-L 0.38 ✓
- **End-to-End Latency** — Full pipeline: < 1.5s avg ✓

Evaluation Methodology

Evaluation uses a held-out test set of 15% of total data, stratified by class label to maintain distribution balance. A separate hard-case set of ambiguous queries and mixed-emotion messages probes robustness beyond clean benchmark performance. Latency is measured on the deployed Cloud Run prototype rather than local Colab inference, capturing realistic network and cold-start effects. Macro F1 is reported alongside accuracy for imbalanced classes such as the 27 Bitext intents, confusion matrices were generated for all classification tasks, and OOS recall is prioritized over precision since a safer chatbot fallback is preferable to a confidently wrong response.

Algorithm Comparison vs. Baselines

Intent: Qwen2.5 vs BERT → +11.1pp over rule baseline
Sentiment: Phi-3.5 vs DistilBERT → +9.5pp over DistilBERT
Urgency: Gemma vs keyword rules → Critical recall 84.7%
NER: GLiNER vs spaCy → Zero-shot, no labeling needed

Baseline Configurations

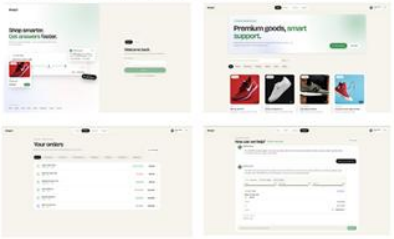
Baselines for fair comparison include a BERT-base intent classifier pretrained on Bitext without QLoRA, a DistilBERT sentiment model with standard full-precision fine-tuning, regex-based keyword urgency rules using terms like "urgent", "asap", and "fraud", and spaCy NER using en_core_web_lg with custom training on a small annotated set. The advantages reported above measure the lift each modern model achieves over its respective baseline on the same held-out test set.

Web Application (SHOP9 demo)

1. Login & Landing — customer auth flow
2. Order Dashboard — history & status tracking
3. Support Chatbot — live NLP, escalation alerts
4. Analytics Panel — sentiment trends, urgency heatmap, BigQuery logs

Live UI Capabilities

The dashboard uses WebSocket-style polling to stream real-time NLP predictions back to the chat interface, with conversation history persisting per session so users can return mid-conversation. Order details open in a modal showing status and item-level breakdown, while a sentiment trend chart visualizes the last 50 queries to give support managers a rolling view of customer mood. Confidence indicators render next to each prediction so human reviewers can spot low-certainty cases at a glance, and toast notifications fire on critical-urgency escalations. The responsive layout adapts cleanly across mobile, tablet, and desktop viewports for agents working across devices.



Tangible Deliverables

AI / ML: 5 QLoRA fine-tuned LLMs, multi-model orchestration, GLiNER NER, escalation logic
Data: Bronze→Silver→Gold GCS pipeline, PII masking, BigQuery inference logging
Stack: React dashboard, FastAPI backend, SQLite orders, Docker, Cloud Run
Deployment: Containerized prototype on Cloud Run, GitHub sourc e control, CI-ready
Evaluation: Benchmark + hard-case test sets, latency profiling, error analysis
Documentation: Final report (DOCX/PDF), slide deck, demo video, this poster
Repository: GitHub with README, run instructions, model checkpoints, notebooks
Reusability: Modular components extensible to other support domains

Pipeline at a Glance

CUSTOMER QUERY
▼
React Frontend
▼
FastAPI Backend
▼
NLP Pipeline (5 LLMs)
▼
Mistral Response Gen
▼
CONTEXTUAL RESPONSE

Logs streamed to BigQuery • Orders served from SQLite • Datasets versioned in GCS Bronze/Silver/Gold

Summary/Conclusions

All 6 outcome targets met or exceeded

- Multi-model LLM orchestration outperformed rule-based and single-encoder baselines by 7–11 percentage points across intent, sentiment, and urgency.
- Sub-2-second end-to-end latency achieved with five sequential models on the deployed pipeline.
- Production-grade stack delivered: React + FastAPI + Docker + GCS + BigQuery + Cloud Run.
- Modular architecture supports straightforward extension to multilingual and voice-based customer support.
- Limitations: benchmark datasets remain more structured than real e-commerce traffic; live CRM integration and concurrency authentication deferred.

Future Work

Several extensions would carry the platform toward production-scale enterprise deployment. Multilingual support across Spanish, Hindi, and Mandarin would expand global reach. A voice-based interaction layer would open accessibility use cases. CRM integration with Salesforce and Zendesk would close the loop into existing enterprise workflows, and retrieval-augmented generation over a product knowledge base would ground responses in catalog facts.

Key References

[1] Larson et al. (2019). An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction. EMNLP.
[2] Zaratiana et al. (2023). GLiNER: Generalist Model for Named Entity Recognition Using Bidirectional Transformer. arXiv:2311.08526.
[3] Dettmers et al. (2023). QLoRA: Efficient Finetuning of Quantized LLMs. NeurIPS.
[4] Hu et al. (2022). LoRA: Low-Rank Adaptation of Large Language Models. ICLR.
[5] Wolf et al. (2020). Transformers: State-of-the-Art Natural Language Processing. EMNLP System Demos.
[6] Demszky et al. (2020). GoEmotions: A Dataset of Fine-Grained Emotions. ACL.
[7] Xiao et al. (2024). C-Pack / BGE-large-en-v1.5: Packed Resources for General Embeddings. arXiv:2309.07597.
[8] Vaswani et al. (2017). Attention Is All You Need. NeurIPS.
[9] Devlin et al. (2019). BERT: Pre-training of Deep Bidirectional Transformers. NAACL.
[10] Sanh et al. (2019). DistilBERT, a Distilled Version of BERT. arXiv:1910.01108.
[11] Microsoft (2024). Phi-3.5-mini-instruct. Hugging Face.
[12] Mistral AI (2024). Mistral-7B-Instruct-v0.3. Hugging Face.
[13] Google DeepMind (2025). Gemma-3-1B-it. Hugging Face.
[14] Qwen Team, Alibaba (2024). Qwen2.5-3B-Instruct. Hugging Face.
[15] Bitext (2023). Customer Support LLM Chatbot Training Dataset. Hugging Face.

Acknowledgements

We thank Dr. Simon Shim and Dr. Lee C. Chang for their guidance throughout DATA 298A and DATA 298B, and the SJSU Department of Applied Data Science for academic and infrastructure support.

We gratefully acknowledge the open-source NLP community; Hugging Face, PyTorch, and the authors of Qwen2.5, Phi-3.5, Gemma, Mistral, GLiNER, and BGE.

We thank Google Colab for providing the GPU compute used during model fine-tuning and evaluation, and Google Cloud Platform for the storage, BigQuery, and Cloud Run resources that hosted the deployed pipeline.

We are grateful to the DATA 298A and DATA 298B cohort for thoughtful feedback during demo reviews, and to the SJSU Applied Data Science faculty for the course framework that made an end-to-end capstone of this scope possible.

Finally, the team members acknowledge each other for the dedication, collaboration, and shared problem-solving that carried this project from initial scoping through final deployment. This work is the product of everyone's effort.

AI-Driven Autonomous Career Mentorship & Market Analyzer

Applied Data Science Department

Project Advisor: Simon Shim, Lee Chang

Bindu Madhavi, Edara
Harika, Eadara
Hemanvitha, Katakam
Sai Swetha, Madapati
Sri Gopi Sarath, Gode

Introduction

Job seekers often receive career advice that is generic, outdated, or not aligned with real job-market expectations. CareerMentor AI analyzes resumes, identifies skill gaps, recommends learning paths, and provides personalized AI mentorship.

The system guides users through RESUME_UPLOAD, SKILL_ANALYSIS, LEARNING_PATH, PROGRESS_TRACKING, and AI_MENTOR_SUPPORT.

Problem Statement

- Job seekers need a clear way to measure their readiness for target roles.We target the gap between **resume skills** and **real job-market expectations**.
- Current platforms give job listings or generic advice, but do not show precise missing skills.
- Learners often lack a structured path for what to learn, practice, and track next.
- CareerMentor AI aims to provide personalized career guidance through resume analysis, skill-gap detection, learning recommendations, and AI mentor support.

Methodology

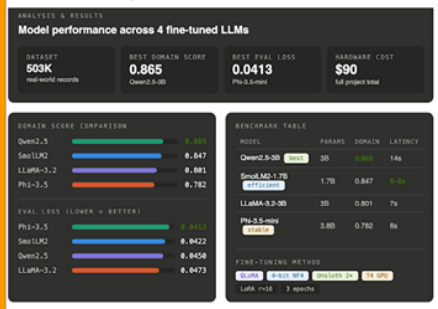
- Collected real-world data from job boards, course platforms, and resume datasets
- Cleaned and structured data into a skill taxonomy and career tracks
- Generated career Q&A pairs to teach the AI domain knowledge
- Fine-tuned four language models from Alibaba, Meta, Microsoft, and Hugging Face
- Evaluated models on response accuracy, quality, and speed
- Built a full-stack web platform with eight career mentorship features
- Deployed a live AI career mentor freely accessible to job seekers

Analysis and Results

1. System Architecture

Tier 1 - User Layer: Job seekers access the platform through a web browser on desktop or mobile.
Tier 2 - Frontend: Netlify hosts the React + TypeScript interface for resume upload, dashboards, learning paths, quizzes, mentor chat, and model comparison.
Tier 3 - Backend: Render hosts the FastAPI service that manages resume parsing, skill-gap analysis, learning resources, progress tracking, quizzes, and REST APIs.
Tier 4 - AI Inference: Google Colab T4 GPU runs four fine-tuned LLMs for mentor chat and live model comparison.
Data Flow: User requests move from browser → frontend → backend → LLM inference through HTTPS/REST and localtunnel.
Deployment: GitHub CI/CD automatically deploys frontend updates to Netlify and backend updates to Render.

2.Models and Components



- **Qwen2.5-3B achieved the highest domain score (0.865)** - Alibaba's multilingual pre-training improved career-specific understanding above all 3B competitors.
- **Phi-3.5-mini had the lowest eval loss (0.0413) but the lowest domain score (0.782)** - proving eval loss alone is an insufficient predictor of real-world advice quality.

- **SmolLM2-1.7B is the best parameter-efficiency choice** - a 1.7B model matched 3B model quality (0.847) at 6–8s inference, the fastest response time in the study.
- **Pre-training architecture fundamentally shapes fine-tuning outcomes** - all 4 models trained on identical data yet produced distinctly different response styles.
- **503K real-world records ground the advice**- training on LinkedIn, Indeed, Glassdoor postings and Udemy/Coursera courses eliminates the generic advice problem of base LLMs.

3. Visualizations

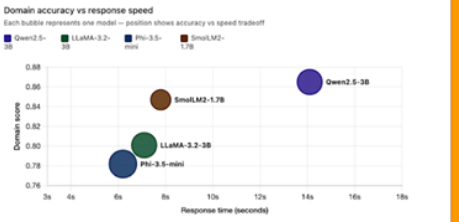


Figure 3.1-Domain Accuracy vs Response Speed

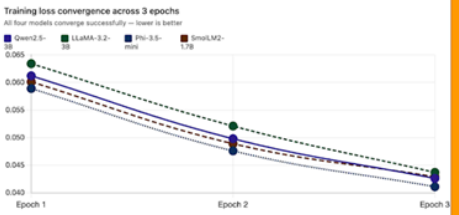


Figure 3.2- Training Loss Convergence Across 3 Epochs

Figure 3.3-Overall Domain Score Comparison

Figure 3.1 shows the tradeoff between domain accuracy and response speed across all four models, where Qwen2.5-3B achieves the highest accuracy but at the slowest response time, while SmolLM2-1.7B delivers competitive accuracy at nearly half the latency making it the optimal choice for real-time career mentorship. Figure 3.2 demonstrates that all four models converged successfully across three training epochs with loss consistently decreasing below 0.045, confirming that the 72,741 career domain training examples were sufficient to adapt each architecture effectively on a single T4 GPU, with Phi-3.5-mini achieving the most stable fine-tuning.

Figure 3.3 compares the overall domain scores evaluated on a held-out test set, where all four models exceed the 0.78 target threshold ,Qwen2.5-3B leading at 0.865, SmolLM2-1.7B at 0.847, LLaMA-3.2-3B at 0.801, and Phi-3.5-mini at 0.782 with the variation in scores despite identical training data confirming that pre-training methodology independently shapes fine-tuned model quality, a key research contribution of this project.

Summary/Conclusions

Our project bridges the gap between job seekers and industry by fine-tuning four large language models on real-world hiring data delivering personalized skill gap analysis, curated learning paths, and AI-powered career guidance in one free platform.

Key outcomes:

- Four LLMs fine-tuned on real job market data using QLoRA
- Skill gap analysis grounded in real hiring patterns
- Eight career features deployed as a live public platform
- Proven that small fine-tuned models rival large commercial AI at near-zero cost

Impact: Democratizing career mentorship data-driven, personalized, and accessible to all.By eliminating the cost barrier of human career coaching, CareerMentor AI empowers every job seeker regardless of background, location, or financial means to compete confidently in today's job market.

Key References

1] Hu et al. (2022). LoRA: Low-rank adaptation of large language models. *ICLR 2022.*

[2] Dettmers et al. (2023). QLoRA: Efficient finetuning of quantized LLMs. *NeurIPS 2023.*

[3] Bai et al. (2025). Qwen2.5 technical report. *arXiv:2501.12599.*

[4] Abdin et al. (2024). Phi-3 technical report. *arXiv:2404.14219.*

[5] Grattafiori et al. (2024). The LLaMA 3 herd of models. *arXiv:2407.21783.*

[6] Allal et al. (2024). SmolLM2: Best language models to the edge. *HuggingFace 2024.*

[7] Li et al. (2025). Skill graph construction from job postings using LLMs. *EMNLP 2025.*

Acknowledgements

We thank our project advisors and the Applied Data Science Department at San José State University for their invaluable guidance, constructive feedback, and continuous support throughout this project. We also thank the open-source ecosystem including Hugging Face, Unsloth, FastAPI, and React for the tools and frameworks that made this project possible.

APPLIED DATA SCIENCE
DEPT | DATA 298B

Guardrail-First Drug Information Assistant

Hallucination Detection & Safe Refusal over Curated DrugBank Evidence

DATA 298B MSDA Project II | San Jose State University | May 2026

TEAM 07

Aayushi Shah · Amaan Khan
Dhruva Joshi · Rohit Dhole
Sneha Guravannavar
Advisor: Prof. Simon Shim

01. INTRODUCTION

THE PROBLEM

LLMs generate authoritative health text even when poorly grounded. Users treat hallucinated output as clinically valid. Existing chatbots optimize for answer rate, not safety.

OUR REFRAMING

We BLOCK when evidence is insufficient. Every response is binary: ALLOW or BLOCK. Fluent output that cannot be justified against curated evidence is refused.

✓ Supported: indications, mechanisms, adverse effects, interactions, toxicity
✗ Blocked: dosing, prescribing, diagnosis, non-drug topics

02. METHODOLOGY

1 Layer 1 Input Guardrail

Every query screened for scope & policy. Dosing, prescribing, diagnosis → blocked before any LLM call, saving cost and preventing risk.

2 BM25 Field-Aware Retrieval

In-scope queries retrieve from 317,410 DrugBank chunks indexed by drug name and semantic type. Transparent, auditable lexical retrieval.

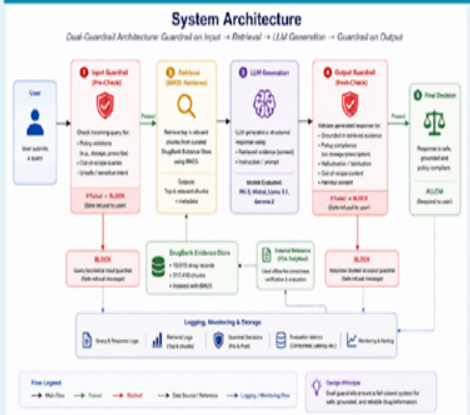
3 Phi-3 Structured Generation

Evidence injected into structured prompt. Phi-3 writes to fixed schema (indication, mechanism, safety). Candidate answer — not yet shown to user.

4 Layer 2 Output Validation

Candidate tested against schema validity, grounding score, and coverage score. Any failure triggers a self-correcting retry before BLOCK.

03. SYSTEM ARCHITECTURE & PIPELINE



Dual Guardrails: Queries are checked before and after LLM generation — blocked at either stage if unsafe, off-policy, or hallucinated.

RAG Pipeline: Approved queries retrieve relevant chunks from a DrugBank Evidence Store via BM25, which ground the LLM's response.

Full Observability: All stages are logged and monitored for decisions, retrieval quality, and evaluation metrics.

04. RESULTS & MODEL COMPARISON

★ Phi-3 = Deployed Default

500-prompt frozen benchmark (350 drug-info, 100 policy, 50 block). All LLMs tested under identical guardrail stack.

Model	ALLOW Rate	BLOCK Correct	Drug-Info ALLOW	Latency (s)	Grounding
★ phi3 (default)	37.4%	100% (150/150)	53.7%	6.3	0.437
mistral	34.8%	100% (150/150)	45.7%	11.0	0.424
llama3.1	33.6%	100% (150/150)	48.3%	19.0	0.418
gemma2	21.4%	100% (150/150)	30.6%	10.0	0.352

Key Finding: All 4 LLMs scored 100% block correctness (150/150) — guardrails enforce safety independent of the generator model. Higher ALLOW = fewer false positives on valid drug queries.

05. DATA ENGINEERING PIPELINE

10,915 records → 317,410 chunks → 35,371 drug names indexed

06. CONCLUSION

Safety Goal Achieved

All four LLMs blocked 100% of policy-sensitive prompts (150/150). Guardrail safety is independent of the generator model used.

Phi-3 Promoted to Default

Highest drug-info ALLOW (53.7%), lowest latency (6.3s), best grounding, perfect block correctness. Selected as deployed chatbot.

Fail-Closed is Correct

In high-stakes domains, refusing safely beats answering confidently. Refusal is a first-class outcome, not degraded UX.

Broader Impact

Pattern applies to compliance assistants, pharmacovigilance, regulated-domain help desks — any AI where unsupported claims cause harm.

07. KEY REFERENCES

[1] Ji et al. (2023). Survey of hallucination in NLG. ACM Computing Surveys, 55(12).

[2] Lewis et al. (2020). RAG for knowledge-intensive NLP. NeurIPS.

[3] Inan et al. (2023). Llama Guard: LLM-based I/O safeguard. arXiv:2312.06674.

[4] Robertson & Zaragoza (2009). BM25 and beyond. Foundations & Trends in IR, 3(4).

[5] Han et al. (2024). MedSafetyBench: Medical safety of LLMs. arXiv:2403.03744.

[6] Wishart et al. (2018). DrugBank 5.0. Nucleic Acids Research.

[7] Sava et al. (2023). NeMo Guardrails. arXiv:2310.10501.

Applied Data Science Department

A Unified XAI Toolkit for Question-Answering Language Models in High-Stakes Domains

Project Advisor: Simon Shim

Chong, Jeff
Ha, Anne
Lee, Jiyoon
Leffler, Matthew
Liu, Nairui

Introduction

As large language models proliferate in high-stakes domains like healthcare, their "black-box" nature creates a critical barrier to trust and adoption. Clinicians and regulators cannot interpret or validate model outputs, and in a domain where a wrong answer can mean a missed diagnosis, this lack of transparency is unacceptable. While XAI research exists, most tools were not designed for medical LLMs and remain inaccessible to non-technical users.

- AI deployed in high-stakes settings requires transparency and accountability
- Without interpretability, practitioners must blindly trust outputs they cannot validate
- Traditional XAI methods were designed/optimized for tabular and vision models, not large language models

Methodology

Toolkit Design and Implementation

A custom MedicalLLMWrapper standardizes inference across four biomedical LLMs of varying scale, integrating four complementary attribution methods: LIME [3], Integrated Gradients [4], TokenSHAP [1], and ELI5 [2], alongside Chain-of-Thought [5] and Best-of-N reasoning methods.

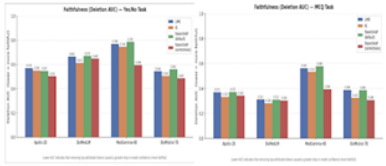
- Custom wrapper enforces constrained decoding and structured outputs across all supported HuggingFace LLMs
- TokenSHAP extended with three custom value functions (correctness-aware, embedding-based, hybrid) and a semantic NER-based splitter
- All methods accessible through a single interactive interface requiring no coding

Analysis and Results

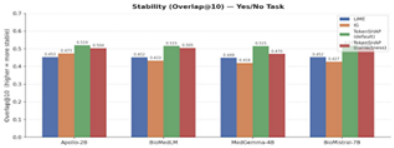
XAI Method Evaluation

Evaluation was conducted across three XAI methods (LIME, Integrated Gradients, and TokenSHAP) over four biomedical LLMs using faithfulness (deletion AUC) and stability (overlap@10) metrics. ELI5 was excluded from quantitative evaluation as its surrogate-based approach produces second-order approximations, making standard metrics unsuitable for direct comparison.

Across both yes/no and multiple-choice tasks, TokenSHAP (correctness) consistently achieved the lowest deletion AUC, indicating the strongest faithfulness overall. Integrated Gradients was competitive across most models, while LIME and TokenSHAP (default) were generally the least faithful. Notably, all methods achieved lower deletion AUC on multiple-choice tasks compared to yes/no, suggesting that attribution methods are better able to identify the tokens driving model decisions in a more constrained answer space. MedGemma-4B showed the highest AUC across both task types, indicating that explanations for larger instruction-tuned models may be harder to faithfully capture.



For stability, TokenSHAP (default) was the most consistent method across all four models, achieving overlap@10 scores between 0.515 and 0.519. The correctness-aware variant was slightly less stable in all cases, pointing to a tradeoff between faithfulness and stability between the two TokenSHAP configurations. LIME showed notably consistent stability across all models at around 0.452, while Integrated Gradients exhibited the most variability, suggesting its attribution behavior is more sensitive to differences in model architecture. Overall, no single method dominated across both faithfulness and stability. This reinforces the value of the toolkit's multi-method design as each method explains model behavior from a fundamentally different perspective, and the appropriate choice depends on the user's priorities rather than a single ranking.



Reasoning Extraction

Chain-of-Thought and Best-of-N sampling were implemented as supplementary reasoning methods to complement the core attribution-based XAI methods. Chain-of-Thought generates a reasoning trace based on the prompt before scoring the answer, avoiding post-hoc rationalization by committing to reasoning before an answer is selected. Best-of-N generates multiple independent reasoning branches, scores each by log-probability, and derives the final answer and rationale from the highest scoring branch. Both methods provide a complementary view of model reasoning that token-level attribution alone cannot capture, though verification currently requires a human in the loop.

User Interface

The toolkit is exposed through a unified Streamlit interface designed to make LLM explainability accessible to both technical and non-technical users. Users can load a model, select a task type, and choose between attribution and reasoning modes without writing a single line of code.

Once a question is submitted, users can run any of the XAI methods to inspect token-level attributions rendered as highlighted text and a ranked bar chart. A cross-method comparison tab aggregates top-k token rankings side by side, giving users a stronger signal of reliability where methods agree.

In reasoning mode, Chain-of-Thought displays step-by-step reasoning traces before committing to an answer.

Best-of-N surfaces multiple reasoning branches scored by log-probability, selecting the highest scoring branch as the final rationale.

Summary/Conclusions

This project delivered a complete, extensible XAI toolkit for auditing and interpreting language model behavior in high-stakes domains. No single method dominated across faithfulness and stability, validating the multi-method design as a comparative framework. Custom QA-aware extensions to TokenSHAP measurably improved faithfulness over the base variant, demonstrating that domain adaptation matters at the explanation level, not just the model level.

Key References

[1] M. Horovicz and R. Goldshmidt, "TokenSHAP: Interpreting large language models with Monte Carlo Shapley value estimation," arXiv, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2407.10114

[2] M. Korobov and K. Lopuhin, "ELI5," Read the Docs, 2021. [Online]. Available: https://eli5.readthedocs.io/en/latest/overview.html

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?": Explaining the predictions of any classifier," arXiv, 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1602.04938

[4] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," arXiv, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1703.01365

[5] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," arXiv, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2201.11903

Acknowledgements

The authors would like to thank Dr. Simon Shim for his guidance and support throughout the project, and Saurabh Suman, Instructional Student Assistant, for providing valuable feedback throughout the project demos. We also thank San José State University for access to the GPU lab and Google Colab academic account credits, which supported the compute-intensive inference and evaluation runs conducted in this project.

Applied Data Science Department

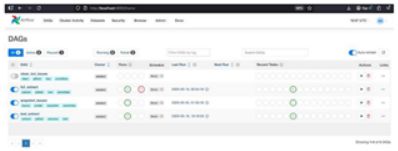
AI-Powered Autonomous Workforce Assistants

Project Advisor: Dr. Simon Shim & Dr. Lee C. Chang

Kadam, Rutuja
Thakur, Shatayu
Mohite, Shreyas
Naik, Shubham

Introduction

Every sprint, engineering teams that manage sizable open-source repositories lose hours due to the same unseen issue: problems accumulate, PRs stop, and by the time a blocker is discovered, a deadline has already passed. Every sprint, the manual triage procedure takes hours and continues to overlook items. The slowest 10% of issues on Apache Airflow alone take more than 22 days to resolve, yet the median closes in a single day. This skew is so severe that traditional alerting criteria completely fail. A Slack-based bottleneck detector and a VSCode autonomous patch planner five refined models with human validation at every stage are two closely integrated AI features developed by AutoBot based on 12,719 actual GitHub issues and 43,731 pull requests.



Methodology

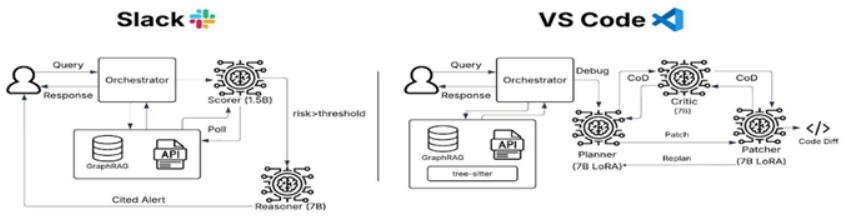
About 56,000 problems and PRs from apache/airflow were used to extract 14 GitHub API signal types using async ETL. Temporally-honest training snapshots were created at project-relative percentiles (P50=1d, P75=5d, P90=22d, and P95=44d), and all closure signals were removed before to training.

Every 30 minutes, the Slack Bot orchestrator uses a Scorer to Reasoner pipeline to identify bottlenecks. It then routes high-risk issues as plain-language alerts straight to the team's Slack channel.

Meanwhile, the VSCode Plugin drives a Planner - Patcher - Critic loop with human approval gates at every stage, ensuring that nothing enters the main branch without developer approval.

This dual-model architecture ensures that critical bottlenecks are addressed with machine precision while maintaining a compact, cost-effective inference footprint. Ultimately, this integration accelerates engineering throughput by reducing the manual overhead of triage for complex, large-scale repositories.

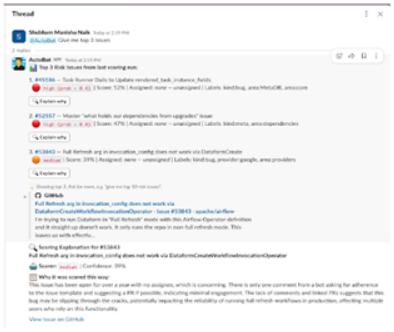
Analysis and Results



Bottleneck Detector:

Every 30 minutes, the Scorer sweeps across all open problems, identifying behavioral cues that humans overlook at scale, such as blocked PRs, assignee changes, comment gaps, and CI failures. It then gives each issue a calibrated risk float that indicates if it requires immediate attention.

Every issue that the Scorer flags as high-risk is translated by the Reasoner into a two to three sentence plain-language briefing that is sent straight to the team's Slack channel. This briefing tells the scrum master exactly what is stalling, why it matters, and how urgent it is without requiring them to open a single GitHub tab.

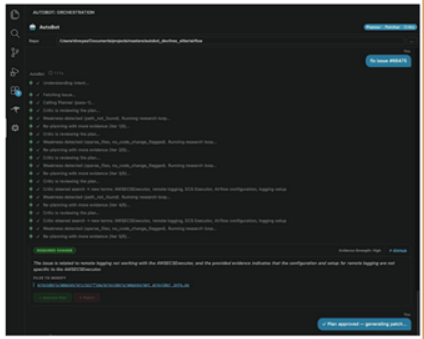


Autonomous Patch Planner:

To determine precisely which files and functions need to be changed, the Planner examines both the problem and the actual codebase structure. This ensures that no paths are misdirected or hallucinated by basing each recommendation on the actual repository.

Using the Planner's results, the Patcher creates a real unified diff that targets the precise function spans found. This is a ready-to-apply code modification rather than a recommendation or explanation of what has to be done.

The developer receives a patch that has already undergone self-review before they see it because the Critic evaluates the generated patch and provides an ACCEPT, REVISE, or REJECT decision with specific line-level reasoning. This process drives an iterative retry loop that detects bugs and architectural violations before CI even runs.



Summary/Conclusions

Summary

AutoBot is a multi-model agentic system that automates software delivery on apache/airflow through two pipelines: a Slack bot that scores open issues for delivery risk and pushes plain-language briefings to stakeholders, and a VS Code extension that autonomously plans, generates, and validates code patches with human-in-the-loop approval. Five LoRA-adapted models (Scorer, Reasoner, Planner, Patcher, Critic) are trained on 12,000+ issues and 41,000+ PRs, backed by hybrid retrieval combining GraphRAG co-resolution patterns with Tree-sitter structural indexing.

Conclusions

- *Hybrid retrieval (GraphRAG + Tree-sitter) outperforms keyword heuristics, historical co-resolution patterns find relevant files while symbol-level spans keep context bounded.*
- *Deterministic orchestrator refinement beats unconditional multi-pass prompting; remaining Planner errors are recall misses best fixed by enriching evidence, not adding LLM calls.*
- *Live RLHF via Slack reactions provides a zero-effort annotation signal, enabling weekly DPO retraining without dedicated labeling infrastructure.*

Key References

[1] Jimenez et al. (2024). SWE-bench: Can language models resolve real-world GitHub issues? ICLR 2024.

[2] Xia et al. (2024). Agentless: Demystifying LLM-based software engineering agents. arXiv:2407.01489.

[3] Hu et al. (2021). LoRA: Low-rank adaptation of large language models. arXiv:2106.09685.

[4] Sayagh et al. (2024). What makes a GitHub issue ready for Copilot? arXiv:2512.21426.

[5] Yang et al. (2024). Qwen2 technical report. arXiv:2407.10671.

Acknowledgements

The authors thank Dr. Simon Shim and Dr. Lee C. Chang, Department of Applied Data Science, San Jose State University, for their guidance throughout the project lifecycle.

Multi-Agent Decision Support System for Retail Operations

Applied Data Science Department

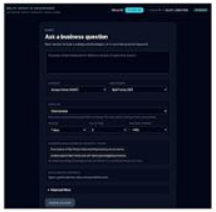
Project Advisor: Simon Shim

Dumaslan, Victor
Han, Dongmei
Saraf Srinivas Rao, Niranjan Rao
Tien, Thi Thao Thien

Introduction

Retail operators face significant cognitive load when making decisions around pricing, inventory, and demand. Most existing AI tools present surface-level information without actionable guidance - working *for* humans rather than *with* them. This results in decision fatigue, missed opportunities, and suboptimal outcomes in fast-moving retail environments.

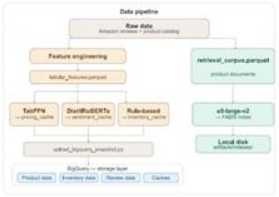
- Operators need AI that reduces cognitive load while preserving human authority over final decisions
- Current tools offer fragmented analytics with no mechanism for structured multi-perspective reasoning
- Our system targets three core areas: pricing optimization, inventory positioning, and demand signal interpretation



Methodology

Data Collection & Pipeline

We sourced data from the Amazon Reviews 2023 dataset (McAuley Lab), focusing on the Home & Kitchen category. Raw JSONL files were cleaned and validated using quality gates - requiring valid product identifiers, ratings between 1.0-5.0, minimum 30-character reviews, and at least 15 reviews per product. Price outliers were handled via 1st-99th percentile winsorization. The final curated dataset contains 50,000 products and 5.3 million reviews, split 70/15/15 into train, validation, and test sets



- Raw JSONL data ingested and validated before storage in Google BigQuery
- Reviews and metadata transformed into Parquet format for downstream agent use
- Synthetic inventory data generated to simulate stock and sales signals for the Inventory Agent

Analysis and Results

To support fast runtime performance, specialist agent outputs - including pricing, sentiment, and inventory signals - were precomputed and stored as caches in BigQuery. The retrieval index and embedding model are loaded into memory at startup, eliminating per-request re-encoding

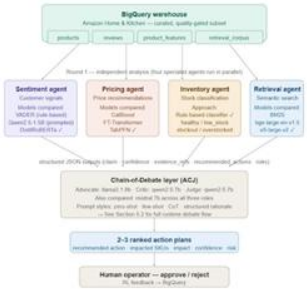
Agent Design & Chain-of-Debate Framework

Specialist agents analyze each retrieved product in parallel before passing structured outputs to the Advocate-Critic-Judge (ACJ) debate layer. Each agent underwent model comparison to identify the best performing approach.

- **Retrieval Agent**: Compared sparse vs dense retrieval
- **Sentiment Agent**: Compared rule-based, prompted LLM, and fine-tuned encoder
- **Pricing Agent**: Compared gradient-boosted, deep learning, and in-context tabular inference
- **ACJ Debate Layer**: Evaluated 27 combinations of LLMs across Advocate, Critic, and Judge roles

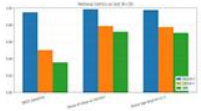
Agent	Models Compared	Evaluation Metrics
Retrieval	BM25, bge-large-en-v1.5, e5-large-v2	Recall@10, nDCG@10, MRR
Sentiment	VADER, Qwen2.5-1.5B, DistilRoBERTa	Macro F1, MCC, Accuracy
Pricing	CatBoost, FT-Transformer, TabPFN	RMSE, MAE, Violation Rate
ACJ Layer	Llama3.1-8B, Qwen2.5-7B, Mistral-7B	Conflict, Justification, Constraint, Latency

In the ACJ debate layer, the Advocate proposes candidate actions, the Critic identifies risks and conflicts, and the Judge synthesizes the final 2-3 ranked action plans. Operators retain final approval authority, with all decisions logged to BigQuery for reinforcement learning feedback.



Retrieval Results

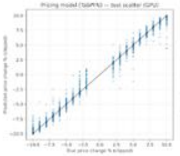
The dense retrieval model e5-large-v2 outperformed both the BM25 baseline and bge-large-en-v1.5 across all metrics, confirming that semantic retrieval provides stronger candidate ranking than keyword-based search. High retrieval quality is critical for the system since all downstream agents depend on the retrieved candidate set.



Model	Recall@10	nDCG@10	MRR
BM25	0.9950	0.9503	0.9357
bge-large-en-v1.5	0.9977	0.9775	0.9707
e5-large-v2 ✓	0.9985	0.9786	0.9720

Pricing & Sentiment Results

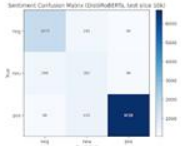
TabPFN outperformed both CatBoost and FT-Transformer on the pricing task within the ±10% policy bound, demonstrating that in-context tabular inference generalizes better than traditional tree ensembles on this dataset



Model	RMSE	MAE	Violation Rate
CatBoost	0.8595	0.2647	0.4372
FT-Transformer	0.9030	0.2391	0.7876
TabPFN ✓	0.4285	0.1435	0.2291

The fine-tuned DistilRoBERTa outperformed both VADER and prompted Qwen across all classification metrics, performing strongly on positive and negative classes. However, the neutral class showed lower precision, which is a known challenge in sentiment classification due to class ambiguity.

Model	Macro F1	MCC	Accuracy
VADER	0.5354	0.4085	0.7284
Qwen (prompted)	0.5298	0.5484	0.7432
DistilRoBERTa ✓	0.7438	0.7343	0.8783



ACJ Results

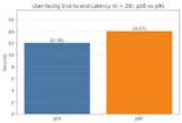
The selected ACJ configuration (Llama 3.1-8B as Advocate, Qwen 2.5-7B as Critic and Judge) achieved the best performance across all rubric metrics while maintaining the lowest latency among all tested combinations. All outputs were valid and correctly structured with an ok_rate of 1.00. A prompt-style ablation further revealed that few-shot prompting improved constraint respect at the cost of conflict detection.

Advocate	Critic	Judge	Conflict	Justification	Constraint	Latency
llama3.1:8b	qwen2.5:7b	qwen2.5:7b	0.90	0.97	0.88	7.63s
mistral:7b	qwen2.5:7b	mistral:7b	0.90	0.83	0.78	8.98s
mistral:7b	qwen2.5:7b	qwen2.5:7b	0.80	1.00	0.73	8.60s
llama3.1:8b	qwen2.5:7b	mistral:7b	0.80	0.80	0.92	8.30s

System Performance

The end-to-end system was evaluated across 20 test queries, achieving a median latency (p50) of 12.08 seconds and p95 of 14.07 seconds. An ok_rate of 1.00 was maintained across all queries, confirming consistent and constraint-compliant outputs.

Agent	Selected Model
Retrieval	e5-large-v2
Sentiment	DistilRoBERTa
Pricing	TabPFN
Inventory	Rule-based
Advocate	Llama 3.1-8B
Critic/Judge	Qwen 2.5-7B



Summary/Conclusions

This project implemented a multi-agent decision support system for retail operations using a Chain-of-Debate (CoD) framework, combining semantic retrieval, sentiment analysis, pricing, and inventory reasoning while preserving human control over final decisions.

Model comparisons confirmed e5-large-v2, DistilRoBERTa, and TabPFN as the best performing models. The ACJ debate layer achieved an ok_rate of 1.00 across all queries. Key limitations include hardware-dependent latency and reliance on synthetic inventory data.

Key References

[1] Liang et al. (2023) - Encouraging Divergent Thinking in Large Language Models Through Multi-Agent Debate

[2] Estornell & Liu (2024) - Multi-LLM Debate: Framework, Principals, and Interventions

[3] Ling et al. (2024) - MeMAD: Structured Memory of Debates for Enhanced Multi-Agent Reasoning

[4] Zhang et al. (2025) - If Multi-Agent Debate is the Answer, What is the Question

[5] Zhai et al. (2025) - Beyond Retrieval-Ranking: A Multi-Agent Cognitive Decision Framework for E-Commerce Search

[6] McGrath et al. (2025) - Collaborative Human-AI Trust (CHAI-T): A Process Framework for Active Management of Trust in Human-AI Collaboration

Acknowledgements

We would like to thank Professor Simon Shim for his guidance and support throughout this project.

We also thank the Applied Data Science Department and the university GPU lab for providing the computational resources necessary for model training and evaluation.

Applied Data Science Department

Prism

Project Advisor: Shim, Simon

Gaddam, Pranav Reddy
Srivastava, Pulkit
Shinde, Shweta
Patil, Gayatri

Introduction

Large language models can solve complex math problems — but Chain-of-Thought explanations only tell us what the model *claims* to compute, not what it *actually* computes internally.
Mechanistic interpretability examines the real forward pass: attention weights, activation magnitudes, and probability distributions at every transformer layer — making the model's internal reasoning visible.
Prism makes this visible across four math-specialized 7B LLMs simultaneously, on the same prompt, in real time.

The platform answers:
Which input tokens drive the final answer?
Where does the model hesitate (low-confidence tokens)?
At which layer does the correct answer first emerge?
Does changing one number flip the model's conclusion?

Methodology

System Architecture
Three-layer design:
Frontend — React 19 + Vite + TypeScript SPA. User submits a math prompt; results appear in a tabbed 4-model bento grid with expandable visualization cards
Backend — FastAPI proxy (no local models). Receives requests from the frontend and forwards them to the GPU server
RunPod GPU Server — Loads all 4 models in VRAM.
Serves /generate, /explain/*, and /posthoc/* endpoints

Data Flow
User Prompt → React Frontend → FastAPI Backend → RunPod GPU → /generate + /explain/* + /posthoc/*

Tech Stack
Frontend: React 19, TypeScript, Vite, Tailwind CSS
Visualizations: D3.js (attribution graph), Recharts (charts), SVG (token flow)
Backend: FastAPI (Python), stateless proxy
GPU Server: PyTorch, HuggingFace Transformers, bfloat16 precision
Deployment: RunPod A100 80GB, ngrok HTTP tunnel
4 × 7B bfloat16 models ≈ 56GB peak VRAM fits on one 80GB instance

Key Engineering Decisions
_GEN_LOCK serializes all 4 parallel frontend requests on a single GPU with zero OOM crashes
No models loaded on the orchestration backend (clean separation of concerns)
Post-hoc analysis is lazy-loaded — only triggered when the user switches tabs

Analysis and Results

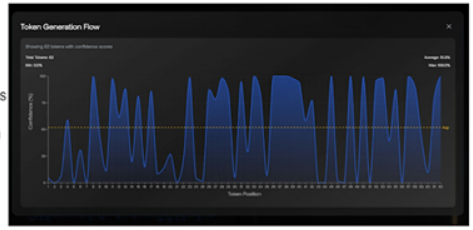
Mechanistic Interpretability ***(during the forward pass)***
Token Confidence
Per-token softmax probabilities over the output vocabulary.
Low-confidence spans reveal exactly where the model is uncertain mid-generation. Displayed as an SVG line chart with a dashed average-confidence reference line and token labels.

Hidden State Norms
L2 norm of hidden activations averaged across the sequence, plotted per transformer layer. Norm growth patterns differ across model families and correlate with reasoning depth. Expandable modal shows min/max/average statistics.

Logit Lens
Projects each intermediate hidden state through the final LM head at every layer. Traces how the model's "best guess" for the next token evolves from early-layer noise to the correct answer — typically crystallizing around layer 16–20 of 32.

Gradient Attribution (Attention Rollout)
Computes input token importance by multiplying and normalizing attention matrices across all layers. Identifies which prompt tokens (numbers, operators, keywords like "total" or "divided") most influence the final answer.

Post-Hoc Analysis ***(after generation completes)***
LIME — Local Interpretable Model-Agnostic Explanations
Masks random word subsets (16–24 samples per prompt), re-generates responses for each, then fits a weighted ridge regression to attribute answer correctness to each word. Implemented entirely from scratch in NumPy — no sklearn dependency. R^2 fit quality score is reported.



TokenSHAP — Shapley Value Attribution
Estimates each word's marginal contribution to correctness via Monte Carlo coalition sampling. Permutes word orderings across samples to compute fair Shapley-value credit allocation. Uses a coalition value cache to avoid redundant forward passes.

Model	Specialty
Qwen 2.5 Math 7B	Alibaba math reasoning
DeepSeek Math 7B	Instruction-following math
InternLM2 Math+ 7B	Step-by-step reasoning
WizardMath 7B V1.1	Fine-tuned problem solving

Counterfactual Editing
Identifies numeric tokens in the prompt (e.g., "12 apples", "5 boxes") and perturbs them (+1 / −1 / ×2). Re-generates responses for each perturbation and checks whether the final answer flips. Finds the minimum numeric change that breaks the model's conclusion.

Attribution Graph (Interactive DAG)
Builds a 3-layer directed acyclic graph: top-8 input tokens (by attention rollout score) → sampled attention layers → output position predictions. Rendered with D3.js force layout. Interactive sliders control how many layers and output tokens to display. Clicking any node shows the top-5 and bottom-5 token predictions at that position.

A Comparison tab displays a side-by-side metrics table across all four models token count, average confidence, max attribution score, average hidden state norm, and logit lens layer count — with the best value per metric highlighted. And since not everyone is a machine learning expert, we added a plain-English AI Interpretation box that automatically explains what each result actually means, so anyone can understand what the model was doing without knowing any math or AI.

Summary/Conclusions

Technical Contributions

- First platform to compare mechanistic interpretability across four math-specialized LLMs simultaneously on the same prompt
- Custom LIME, TokenSHAP, and counterfactual implementations entirely in NumPy/PyTorch — no sklearn or SHAP library
- Interactive attribution graph (D3) with live layer/output sliders, built from real attention rollout data
- GPU scheduling enabling 4 × 7B models on a single 80GB instance with zero OOM crashes

Key Findings

- Models with steeper hidden-state norm growth in middle layers (12–20) produce more structured, step-by-step reasoning traces
- Logit lens reveals correct answers often crystallize as early as layer 16/32 later layers refine confidence rather than change the prediction
- Counterfactual analysis shows DeepSeek Math is most sensitive to numeric perturbations
- LIME attribution consistently highlights the final numeric value and operation keyword ("divided", "total") as the top-attributed words in the prompt

Key References

[1] Elhage et al. (2021). *A Mathematical Framework for Transformer Circuits*. Anthropic Technical Report.
[2] Ribeiro, Singh & Guestrin (2016). *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. ACM KDD.
[3] Lundberg & Lee (2017). *A Unified Approach to Interpreting Model Predictions*. NeurIPS.
[4] Nostalgebraist (2020). *Interpreting GPT: the Logit Lens*. LessWrong.
[5] Wiegreffe & Pinter (2019). *Attention is not Explanation*. EMNLP.
[6] Abnar & Zuidema (2020). *Quantifying Attention Flow in Transformers*. ACL.

Acknowledgements

Thanks to RunPod for GPU infrastructure access, and to the Applied Data Science faculty for guidance throughout this project. Special thanks to the open-source communities behind HuggingFace Transformers, React, D3.js, and PyTorch.

Applied Data Science Department

GuardRail: Hallucination Detection and Prevention in LLMs

Project Advisors: Dr. Simon Shim & Dr. Lee C. Chang

Hayagreevan, Sreenidhi - 018195489
Beri, Shivani Atul - *018205018*
Kalvakota, Laxmi Thrishitha - *017605640*
Meravanige Veerappa, Bhavana - *017852458*

Introduction

Large Language Models (LLMs) are powerful but often generate hallucinated, unsafe, or policy-violating responses, making them unreliable for real-world applications. They also remain vulnerable to prompt injection and adversarial inputs, with limited mechanisms for enforcing safety in real time.

To address this, we developed GuardRail for LLM, a modular safety framework that acts as an intermediate layer between users and the model. The system incorporates input validation, prompt control, and output filtering to detect and block harmful or irrelevant content.

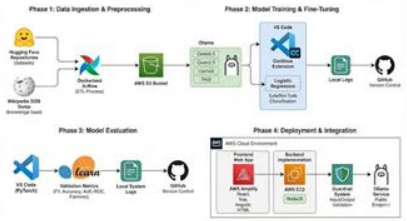
By enforcing structured constraints at both the input and output stages, GuardRail significantly improves response reliability, safety, and consistency, enabling more trustworthy deployment of LLM-based applications.

Methodology

We implemented **multi-layer guardrail system** combining machine learning, rule-based filtering, and retrieval-augmented validation (RAG) to enforce safety. A **knowledge base (Wikipedia + domain text)** was built, preprocessed and chunked (~600 chars), and embeddings were generated using SentenceTransformer for semantic search.

For input safety, we trained a **binary classifier (Logistic Regression)** on text embeddings to classify safe vs unsafe prompts. The model is stored as a **.joblib** for real-time inference and captures contextual unsafe intent beyond keyword matching. This is complemented by **regex filters** for PII, hate speech, bias, violence, and fraud. The **RAG pipeline** retrieves relevant chunks to provide context grounding and reduce hallucinations. The LLM (Ollama) processes only validated inputs using retrieved context and structured prompts.

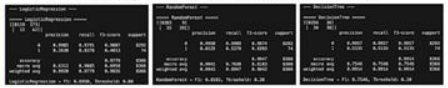
A post-processing layer applies **output guardrails**, including hallucination and PII detection. Unverifiable responses are replaced with a safe fallback. The system is optimized using a shared embedding model, similarity threshold (~0.3), background threading, and modular middleware design.



Analysis and Results

The GuardRail system was evaluated across **safety classification, hallucination detection, and end-to-end system validation** using benchmark datasets and real-time deployment tests.

For **input safety classification**, multiple models were compared. **Logistic Regression achieved 97.79% accuracy with high recall (0.8378)**, ensuring most unsafe prompts are detected, though with some false positives. **Random Forest achieved the best overall performance (99.47% accuracy, F1 = 0.8183)**, providing a balanced trade-off between precision and recall, while Decision Tree showed moderate performance.

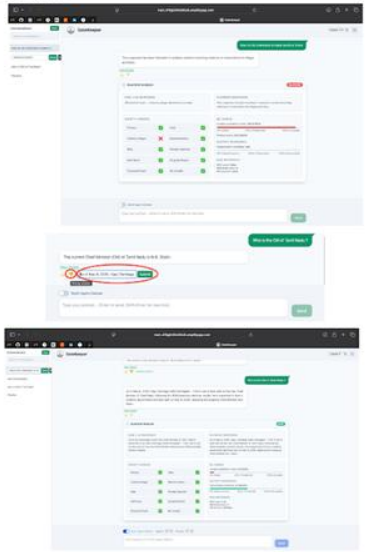


The system was validated using **7 functional test cases** (unsafe prompts, hallucination correction, bias detection, privacy protection), all resulting in **PASS**, confirming reliable end-to-end behavior.

The **GateKeeper UI** provides full transparency of guardrail decisions. As shown, unsafe prompts (e.g., harmful queries) are **blocked at input stage**, with the interface displaying:

- **Raw LLM response (blocked before execution)**
- **Guarded response (policy-compliant fallback)**
- **Safety checks across categories (PII, bias, violence, etc.)**
- **ML unsafe probability score (~96%)**
- **RAG metadata and hallucination score**

This makes the system **explainable and auditable**, unlike black-box LLM outputs.



The **RAG-based knowledge layer** improves factual reliability by retrieving context from structured datasets and Wikipedia. Incorrect LLM outputs are **corrected using retrieved evidence**, reducing hallucinations and ensuring grounded responses.

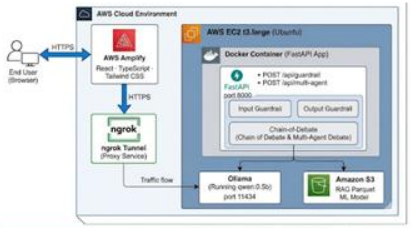
The **output guardrail layer** further validates responses using **context similarity and PII detection**, blocking or replacing unsafe outputs with safe fallbacks.

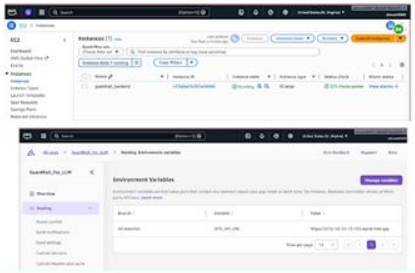
Overall, the system achieves:

- **High recall in unsafe prompt detection**
- **Effective hallucination reduction via RAG grounding**
- **End-to-end safety enforcement (input + output layers)**
- **Explainable decisions through interactive UI**

Cloud Deployment

The system is deployed as a **full-stack cloud application**. The frontend (**React + TypeScript**) is hosted on **AWS Amplify**, while the backend (**FastAPI**) runs on an **AWS EC2 instance** with Dockerized services. The LLMs are served locally via **Ollama**, and datasets, embeddings, and model artifacts are stored in **Amazon S3**. API communication is secured via HTTPS (ngrok tunneling), enabling a scalable, modular, and real-time guardrail system suitable for enterprise deployment.





Summary/Conclusions

This project presents a **multi-layer GuardRail framework** for Large Language Models that integrates **input validation, RAG-based grounding, multi-model reasoning, and output verification** to address key challenges such as unsafe content generation and hallucinations.

The system combines **ML-based safety classification, rule-based filtering, and retrieval-augmented validation** to enforce policy compliance and factual accuracy. Evaluation across benchmark datasets and real-time test cases shows that the framework effectively **blocks unsafe prompts, reduces hallucinations, and produces context-grounded responses**while maintaining low latency.

The **GateKeeper UI** enhances transparency by exposing guardrail decisions, including safety checks, ML scores, and response validation, enabling **explainable and auditable AI behavior**.

This project is beneficial as it provides a **reliable safety layer for deploying LLMs in real-world applications**, reducing risks associated with harmful content, misinformation, and bias. By ensuring **trustworthy, verifiable, and policy-compliant outputs**, the system supports safer adoption of AI in **high-stakes domains such as healthcare, finance, and legal systems**, where accuracy and accountability are critical.

Overall, the framework offers a **scalable, modular, and enterprise-ready solution** that advances LLM reliability by combining **detection, grounding, and verification** into a unified pipeline.

Key References

[1] Lin, S., Hilton, J., & Evans, O. (2021). TruthfulQA: Measuring How Models Mimic Human Falsehoods. ACL. https://arxiv.org/abs/2109.07958

[[2] Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS. https://arxiv.org/abs/2005.11401

[3] Rajpurkar, P., et al. (2016). SQuAD: 100,000+ Questions for Machine Comprehension. EMNLP. https://rajpurkar.github.io/SQuAD-explorer/

[4] Bowman, S., et al. (2015). A Large Annotated Corpus for Learning Natural Language Inference (SNLI). EMNLP. https://nlp.stanford.edu/projects/snli/

[5] Williams, A., et al. (2018). MultiNLI: A Multi-Genre Natural Language Inference Corpus. NAACL. https://cims.nyu.edu/~sbowman/multinli/

[6] Davidson, T., et al. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. ICWSM. https://arxiv.org/abs/1703.04009

Acknowledgements

We express our gratitude to our project advisors, **Dr. Simon Shim**, for guidance throughout this project. We thank the **Department of Applied Data Science, SJSU**, for providing all academic resources. We acknowledge the open-source community (Hugging Face, PyTorch, Ollama, Apache Airflow, scikit-learn) and the benchmark datasets used, including TruthfulQA, SQuAD, Safety Prompts, Hate Speech, and NLI datasets.

Applied Data Science Department

Project Echo:

An AI Music Generation Platform Using Multi-Architecture Transformer and Latent Diffusion Systems with Retrieval-Augmented Guidance

Chauhan, Deeksha
Aggarwal, Moksh
Mahendru, Shivam
Premanand, Rajarajeswari

Project Advisor: Professor Simon Shim

Introduction

Project Echo is a text-to-music generation system that creates instrumental tracks from natural-language prompts. Built within a strict 15-hour A100 GPU budget, it integrates five model architectures alongside a dual-modality Music RAG module over MusicCaps.

Three core components — M3 (11.3M params), M4 (LoKR-adapted ACE-Step 1.5), and M5 (dual-modal RAG) — unified under a session-safe training pipeline.

- **LoKR adaptation** *of ACE-Step 1.5 — 0.03% trainable parameters, outperforming LoRA*
- **Zero-parameter style transfer** *via Attention Intervention and RoPE-enhanced chroma cross-attention,*
- **Dual-modal Music-RAG** *— caption + CLAP retrieval in <500 ms, human-editable CB0 skeleton*

Methodology

Five-Architecture Model Design

Project Echo builds five complementary models across a 15-hour A100 budget, spanning fine-tuning, from-scratch training, parameter-efficient adaptation, and retrieval-augmented generation — all validated with zero decode errors.

- **M1** *— MusicGen fine-tuned on 1,161 EDM tracks via chroma cross-attention (~1 hr A100)*
- **M2** *— ACE-Step + LoKR with style-steerable synthesis via structured text labels (genre, BPM, key, mood)*
- **M3** *— 11.3M param two-stage transformer trained from scratch; CCS 0.896 across 6 genres in ~40 min*
- **M4** *— ACE-Step LoKR (0.03% of 4.8B params); SDD −32.4%, FAD −23.8% in 15.3 GPU-hours*
- **M5** *— Caption + CLAP dual-modal retrieval in <500 ms via FastAPI + ChromaDB*

Analysis and Results

M4 Primary Evaluation — Base vs. Fine-Tuned

The fine-tuned model achieves consistent improvement across all spectral metrics evaluated on 25 generated samples:

Metric	Base	Fine-Tuned	Delta
SDD (lower=better)	0.704 ± 0.604	0.476 ± 0.189	-32.4%
FAD (lower=better)	51.344	39.105	-23.8%
CCS (higher=better)	0.819 ± 0.057	0.812 ± 0.067	-0.8% (expected)

The SDD standard deviation compression from ±0.604 to ±0.189 is significant — it shows the fine-tuned model is not just occasionally better but consistently closer to the reference distribution. The CCS result of −0.8% is within noise and was fully expected: LoKR training targeted the spectral envelope, not harmonic structure. FAD is computed on CLAP mel-spectrogram embeddings, making it sensitive to perceptual audio quality. The 23.8% FAD improvement therefore reflects a meaningful gain in perceived audio naturalness.

M1 Evaluation — Perplexity and Token Diversity

Seed: 50 tokens (≈1 sec) · Generated: 350 tokens (≈7 sec) · Temperature: 0.85 · Top-k: 50

A Fine-tuned on 1,161 NoCopyrightSounds EDM tracks over 3 epochs (~1 hour on A100). Perplexity dropped measurably from the epoch-0 baseline to a lower value at epoch 3, confirming the model learned EDM-specific melodic token distributions. Generated MIDI decoded to coherent melodic patterns with recognizable EDM stylistic features. Token diversity analysis confirmed generated sequences do not collapse to repetitive high-frequency tokens.

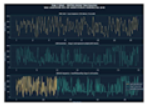
M2 Evaluation — A/B Comparison and CLAP Similarity

Multi-stage LoRA/LoKR fine-tuning over 500–1500 epochs on 20 classical and world music clips. A/B Gradio comparisons at each checkpoint showed progressive style acquisition from structured text descriptions. By epoch 500, generated outputs began reflecting target genre vocabulary in instrument timbre and phrase structure. Drive-based checkpointing was necessary — Colab Pro sessions typically disconnect every 8–12 hours.

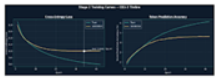
M3 Inference — *Six Genres (seed 50 tokens ≈1s; generated 350 tokens ≈7s)*

Genre	CCS	Clip	Rating
NCS Electronic	0.9295	PASS	Excellent
Ambient	0.8305	PASS	Excellent
Melodic Ambient	0.9040	PASS	Outstanding
Melodic	0.9538	PASS	Outstanding
Chill Synthwave	0.8660	PASS	Excellent
Cyberpunk	0.8331	PASS	Excellent
Average	0.8962	6/6	Excellent

(a) Stage 1 (CB0, 80 ep.). Best val loss 4.5368, perplexity 93.4.



A perplexity of 93.4 (21.9× better than random baseline) confirms genuine melodic pattern learning across 303 source tracks. Average CCS of 0.8962 across six genres shows harmonic character is consistently preserved. Stage 2 learning rate correction from 10^{-3} to 3×10^{-4} resolved premature convergence; all six samples passed the clipping test.

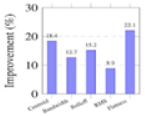


(b) Stage 2 (LR 3×10^{-4}, 34 ep.). Best val loss 5.9981.

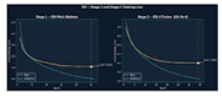
SDD std-dev compression (±0.604 → ±0.189) confirms consistently better output, not occasional gains. The 23.8% FAD improvement reflects a meaningful gain in perceived audio naturalness validated in CLAP embedding space.

LoKR adapter: 2.53 → 0.85 (−66.3%, 30 ep.) · RoPE cross-attention: 0.291 → 0.211 (−27.6%, 200 ep.) · LatCH harmonizer: 1.749 → 0.940 (−46.3%, 100 ep.) · Attention Intervention: 0 parameters, inference-time only.

M5 Music-RAG query "emotional music"*: caption-semantic path returned 54.9% similarity; CLAP audio path returned 61.0% similarity. Both paths returned non-overlapping results; all 5 test queries served within 500 ms.*

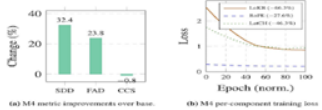


(a) Per-feature spectral gains.



(b) M3 evaluation summary.

SDD: 0.704→0.476 (−32.4%) · FAD: 51.344→39.105 (−23.8%) · CCS: −0.8% within noise. Std-dev compression (±0.604→±0.189) confirms consistently better output.
LoKR: −66.3% · RoPE: −27.6% · LatCH: −46.3% · Attention Intervention: 0 params, inference-time only.



(a) M4 metric improvements over base. **(b)** M4 per-component training loss

Summary/Conclusions

Project Echo *validates multi-architecture AI music generation under academic compute: M3 hit CCS 0.896 in ~40 GPU-minutes, M4 cut SDD/FAD by 32/24% in 15.3 GPU-hours, M5 RAG <500 ms, zero decode errors.*

Contributions*: human-editable mid-generation checkpoint, LoKR outperforming LoRA, zero-parameter style transfer. Limitations: no MOS, 8s cap, LatCH ablation pending, no vocals.*

Key References

[1] Copet et al., "Simple and controllable music generation," NeurIPS 2023.

[2] Agostinelli et al., "MusicLM: Generating music from text," arXiv:2301.11325, 2023.

[3] Evans et al., "Fast timing-conditioned latent audio diffusion," ICML 2024.

[4] ACE-Step Team, "ACE-Step 1.5: Open-source music generation foundation model," 2025.

[5] Defossez et al., "High fidelity neural audio compression," arXiv:2210.13438, 2022.

[6] Hu et al., "LoRA: Low-rank adaptation of large language models," ICLR 2022.

Acknowledgements

We gratefully acknowledge **Prof. Simon Shim** and the **SJSU Department of Applied Data Intelligence** for their mentorship and support throughout Project Echo.

All compute was provided via **Google Colab Pro** (NVIDIA A100, ~$10–12/month), with all training data and pre-trained models sourced from open, permissively licensed repositories.

Applied Data Science Department

WeaveNet: Interactive Hierarchical Custom Built Graph Database

Project Advisor: Shim Simon

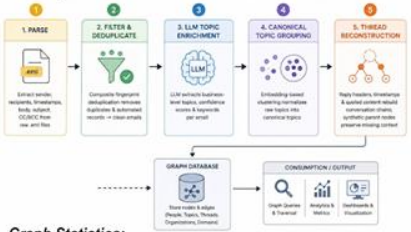
Bhanap, Anvay
Muralidharan, Chetana
Prashant Vichare, Kedar
Alpesh Dedhia, Meera
Prasanna, Mounashree

Introduction

Enterprise email archives hold vast organizational knowledge - yet traditional keyword search fails to reveal *who* collaborates with *whom*, which topics dominate, and how people are connected across the organization. **WeaveNet** transforms the Enron email corpus - 517,401 raw emails from a real corporate network - into a semantically rich, interactive graph database. The system supports communication-pattern discovery, compliance analysis, and organizational intelligence through graph traversal, approximate search, and a React-based visual explorer - going far beyond what keyword retrieval alone can offer.

Methodology

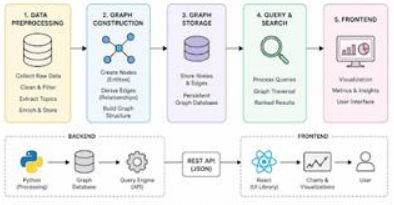
Data Pipeline:



Graph Statistics:

Total Nodes	764,737
Total Edges	2,860,219

System Architecture:



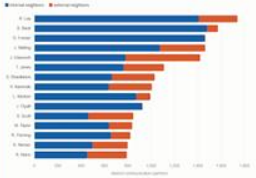
Algorithm Families:

Family	Algorithms
Neighbor Traversal	HNSW vs. NSG
Path Discovery	ScaNN vs. IVF-Flat
Content/Keyword Search	LSH vs. SimHash
Ranking & Similarity	Annoy vs. VP-Tree
Direct Index Lookup	Index-Only

Analysis and Results

Analytics:

Bridge node analysis: Each bar shows internal (enron.com) vs external communication partners. 3,316 persons act as bridges; only top 15 shown.



Algorithm Performance Summary:

Algorithm Pair	Key Metric	Winner Score	Runner-Up Score
HNSW vs NSG	Recall@K	0.943	0.729
HNSW vs NSG	Edge-weight ordering accuracy	0.939	0.840
ScaNN vs IVF-Flat	Path success rate	0.908	0.790
ScaNN vs IVF-Flat	Path completeness	0.949	0.887
LSH vs SimHash	Precision@K	0.858	0.813
LSH vs SimHash	False positive rate	0.142	0.187
Annoy vs VP-Tree	NDCG@K	0.847	0.822
Annoy vs VP-Tree	Recall@K	0.895	0.863
Index Lookup	Hit rate / Sort correctness	1.000	—

Query Latency by Family:

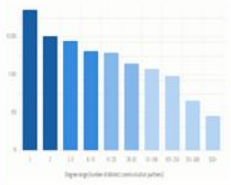
Query Family	Avg. Latency	Suitable For
Index Lookup	**7.3 ms**	Real-time interactive lookup
HNSW Neighbor Traversal	**16.4 ms**	Interactive graph expansion
LSH Content Search	**113 ms**	Analytical content search
ScaNN Path Discovery	**273 ms**	Constrained path analysis
Annoy Ranking	**287 ms**	Top-K ranking & similarity

Neo4j Benchmark Highlights:

Query	Our Engine (p50)	Neo4j (p50)	Speed-up
Neighbor lookup	0.548 ms	3.440 ms	**6.3×**
Path traversal	0.070 ms	5.112 ms	**73×**
Keyword search	4.360 ms	53.597 ms	**12.3×**
Direct lookup	0.063 ms	2.877 ms	**45.7×**
Index lookup	0.072 ms	5.676 ms	**79.4×**

Trade-off: Neo4j is faster for ranking-heavy workloads (Q25, Q26, Q27) - a clear target for future optimization.

No single algorithm dominates all query types, a multi-algorithm architecture is essential.



Evaluation Framework:

The system runs 40 predefined query templates across 5 families, comparing each algorithm against exact ground truth on:

- Correctness: Recall@K · Precision@K · NDCG@K · Path Success Rate · Lookup Hit Rate
- Efficiency: Query Latency (ms) · Nodes Visited · Edges Examined · Memory Usage
- Stability: Rank Stability · Result Fan-out Variance · NDCG across runs
- Frontend: Payload Completeness · Render Correctness · End-to-end Response Time

Web Application UI:

Real-world Use Cases:

- **Enterprise Search:** Retrieve emails by topic, person, or relationship, not just keywords
- **Compliance & Legal Discovery:** Trace communication between people, orgs, and time periods
- **Organizational Network Analysis:** Find communication hubs, collaboration clusters, and info flow patterns
- **AI Assistant Grounding:** Power evidence-backed answers: *"Who discussed this agreement?"*

Summary/Conclusions

WeaveNet successfully transforms a noisy corporate email corpus into a context-aware, semantically enriched graph intelligence system. No single algorithm dominated all query types - HNSW excels at neighbor traversal, ScaNN at path discovery, LSH at content search, and Annoy at ranking - confirming the value of a multi-algorithm architecture. Direct index lookups outperform all approximate methods for exact queries and are significantly faster than Neo4j for those workloads.

Key References

[1] B. Klimt and Y. Yang, "The Enron Corpus: A New Dataset for Email Classification Research," *ECML*, 2004.
[2] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.
[3] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.
[4] Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using HNSW," *IEEE TPAMI*, 2018.
[5] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions via Hashing," *VLDB*, 1999.
[6] M. S. Charikar, "Similarity Estimation Techniques from Rounding Algorithms," *STOC*, 2002.

Acknowledgements

We would like to thank our Project Advisor, Simon Shim, for their continued guidance, feedback, and support throughout the development of this project.

We also extend our gratitude to the Applied Data Science Department for providing the resources, framework, and opportunity to undertake this work.

Evaluating Neural Operator Architectures for Datacenter Digital Twin Surrogate Models

FNO vs U-Net on Large-Scale Structured CFD Data (NVIDIA PhysicsNeMo-Datacenter-CFD)

Sai Ranga Reddy Nukala, Shaila Reddy Kankanala, Sadhvi Singh, Akshith Reddy Jonnalagadda
DATA 298B - MSADI Project II, Spring 2026 | Department of Applied Data Science, San José State University

Motivation & Problem Statement

Datacenters consume **~1.5% of global electricity** (2024), projected to double by 2030 with rising AI workloads. A single hotspot can trigger emergency shutdowns and hardware damage.

CFD simulations (e.g. OpenFOAM) yield high-fidelity thermal predictions but take **up to 15 hours** per geometry - impractical for real-time management.

Neural surrogate models promise a 15,000× speedup, but the critical unanswered question is:

Which neural architecture should power the predictive engine of a Datacenter Digital Twin?

No prior study has evaluated FNO on NVIDIA's public *PhysicsNeMo-Datacenter-CFD* benchmark with comprehensive multi-field evaluation in physical units.

Dataset & Experimental Setup

960	7.37M
CFD Simulations	Grid Points / Sample
15 hrs	**6**
CFD Runtime	Models Benchmarked

PhysicsNeMo-Datacenter-CFD

- 960 structured CFD simulations, grid: **960×96×80**
- 3 physical fields: **T (°C), U (m/s), p (Pa)**
- Input features: coordinates, wall distance, server geometry
- Split: 768 train / 192 test (fixed seed)

Hardware Platform

- **NVIDIA GB10** (Grace Blackwell, 128 GB unified memory)
- Alienware RTX 5080 (prototyping)

Six Models Benchmarked

- Spatial 3D U-Net, PI-U-Net
- Spectral FNO, PI-FNO
- Transformer Transolver
- Baseline POD+MLP

Architecture Overview

✔ 3D U-Net (Recommended)

Params: 22.6M | **Type:** CNN

Key: Encoder–decoder with skip connections preserving fine-scale spatial detail on structured grids

FP16 support: Full (standard convolutions)

Inference: 390 ms FP32 → 236 ms FP16

3D FNO

Params: 28.3M | **Type:** Neural Operator

Key: Global spectral convolutions via FFT; resolution-invariant in theory

FP16 support: cuFFT requires FP32 for non-power-of-2 dims

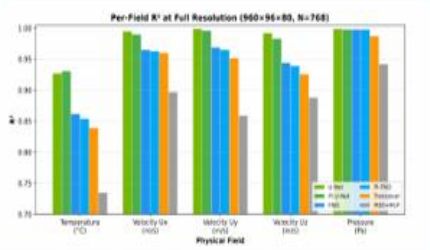
Inference: 680 ms FP32 only

FNO Architecture Diagram

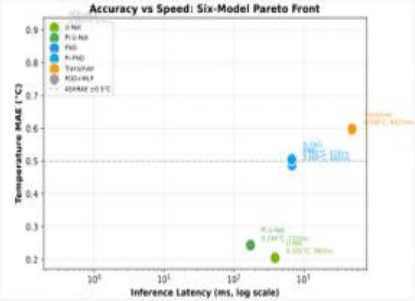
U-Net Encoder–Decoder Architecture Diagram

Results: Full-Resolution Per-Field Accuracy (All 6 Models)

Field	U-Net	PI-U-Net	FNO	PI-FNO	Transolver	POD+MLP
T (°C) MAE	**0.205**	0.244	0.489	0.505	0.598	0.902
T R^2	**0.927**	0.931	0.862	0.854	0.839	0.735
Ux (m/s) MAE	**0.023**	0.034	0.074	0.080	0.077	0.144
Uy (m/s) MAE	**0.027**	0.045	0.143	0.161	0.189	0.342
p (Pa) MAE	**0.068**	0.099	0.099	0.104	0.249	0.573
Latency (ms)	390	**172**	680	674	4972	**0.3**



Per-Field R^2 Bar Chart

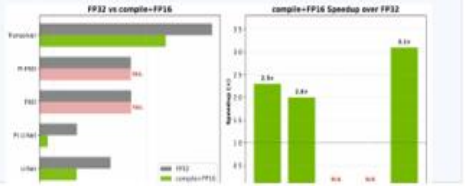


Accuracy–Speed Pareto Front

Key Finding: U-Net achieves **2.4× lower temperature MAE** (0.20°C vs 0.49°C) and **R^2 = 0.927** vs FNO's 0.862.

Key Discovery: Architecture Ranking Reversal

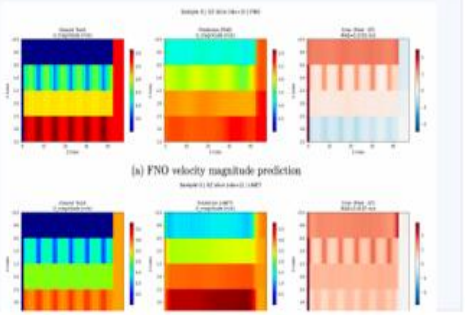
FNO outperforms U-Net at downsampled resolution (×20), but U-Net **dominates at full resolution** - across all fields and all training set sizes.



Inference Optimization & Deployment

PI-U-Net FP16	87 ms 🏆
U-Net FP16	222 ms ✓
U-Net FP32	408 ms
FNO FP32	680 ms
FNO FP16	FAIL

Temperature Prediction Visualization



References

[1] IEA, Electricity 2024, Paris, 2024.
[2] Yue et al., "FNO/U-Net/ViT on datacenter CFD," 2025 (private dataset).
[3] Cao et al., "GNOT for datacenter thermal prediction," NeurIPS 2023.
[4] Rahaman et al., "Spectral bias in neural networks," ICML 2019.
[5] OpenFOAM Foundation, OpenFOAM v11, 2023.
[6] NVIDIA / Wistron, PhysicsNeMo Datacenter CFD, 2024.
[7] Li et al., "Fourier Neural Operator," ICLR 2021.
[8] Herde et al., "Poseidon / Transolver," ICML 2024.

Applied Data Science Department

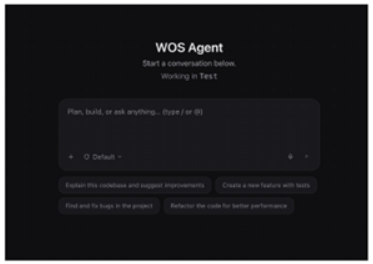
WOS

Project Advisor: Simon Shim

Shah, Jayam
Gangadhar, Thejes Raj
Katipally, Yashwanth Reddy
Chen, Winston

Introduction

Most AI tools live in the cloud. WOS lives on your machine. WOS is a desktop AI agent that reads your files, runs code, connects to your apps, and remembers past conversations -- without sending any data to an external server. No subscriptions, no accounts, no data leaving your computer. It integrates with Slack, GitHub, Jira, and Google out of the box. You can extend it with plain-English Skills and Rules, or plug in any community tool server using the Model Context Protocol standard.

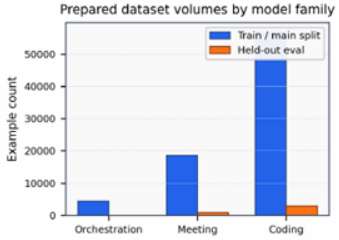


Methodology

We treated reliability as a data problem, not a prompt problem :

1. Generate. We synthesized multi-turn tool-use trajectories that cover Slack, GitHub, Jira, and Google, plus single-step, triage, and cross-app patterns. Errors and clarification turns were injected at calibrated rates so the model learns to recover.
2. Audit and repair. Each trajectory is checked against strict JSON schemas. Bad records are repaired or rejected, never patched silently.
3. Split. Train, validation, and blind-test sets are scenario-disjoint, so evaluation is honest.
4. Fine-tune. Qwen3-32B is trained with QLoRA in 4-bit, then served on a Hugging Face Space behind a vLLM endpoint that speaks the OpenAI tool-calling contract.

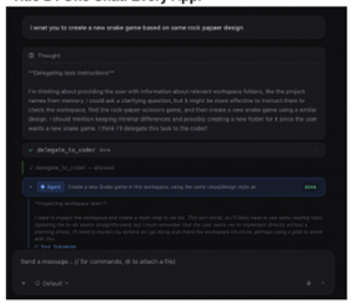
Prepared dataset volumes by model family



Analysis and Results

WOS is an Electron desktop app with a strict main and renderer split. The main process owns orchestration, tool execution, and an encrypted SQLite store. The renderer is a clean React surface for chat, projects, meetings, automations, and apps. A provider layer lets the same conversation run on OpenAI, Anthropic, or our own fine-tuned model with no code change in the UI.

Title 1 : One Chat. Every App.



Structured eval, 27 scenarios across ten behavioral categories:

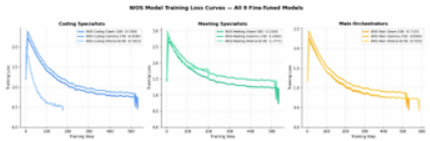
Model	Score	%	Halluc.	Lat. p50 (s)	Lat. mean (s)
wos-orch (fine-tuned Qwen3-32B)	27/27	100%	0	3.24	3.24
Qwen3-32B-bnb-4bit (base, think off)	22/27	81%	0	7.56	8.94
meta/llama-3.3-70b-instruct (NIM)	24/27	89%	0	1.63	4.33

Blind chain eval, 5 multi-step workplace tasks:

WOS fine-tuned	80%
Llama 3.3-70B (NIM)	80%
Qwen3-32B base	0%

The fine-tuned 32B matches the 70B frontier model on real multi-step work, with zero hallucinations.

Title 2 : Specialists for Code and Meetings



Coding specialist:
HumanEval pass@1 80.0% (matches an untuned 70B baseline)
MBPP pass@1 55.0% (matches the same baseline)

Meeting specialist on DialogSum:
ROUGE-1 F1 22.23 (baseline 19.77, +2.46)
ROUGE-2 F1 6.22 (baseline 5.65, +0.57)
ROUGE-L F1 16.65 (baseline 14.43, +2.22)

The meeting model writes summaries that recall more of the source. The coding model holds the line at a fraction of the size.

Title 3: How WOS Thinks

The agent runs a tight loop. Plan the next step. Pick a tool. Check the call against a JSON schema. Run it. Read the result. Decide if the task is done.

Every step is logged. Every tool call is typed. If a tool fails, the loop sees the error and tries a different path. The user always sees what was done and why.

Title 4: Works in the Wild

These five multi-step tasks were never seen during training. WOS ran each one end to end, choosing the right tools and the right order without guidance.

Task	WOS	NIM (70B)	Base
Incident response triage	100%	100%	0%
Sprint planning prep	83%	100%	0%
Weekly cross-app digest	83%	67%	0%
New feature kickoff	100%	100%	0%
Stale PR triage	33%	33%	0%
Average	80%	80%	0%

Each task is a 3 to 4 step chain across Slack, GitHub, and Jira. The base model scores zero on every chain. WOS and the 70B frontier model are neck and neck, at one-twelfth the size.

The hardest task (PR triage, 33%) tripped both WOS and the frontier model the same way: both retrieved the pull requests correctly, then asked for clarification instead of filing the Jira ticket. Same failure, same root cause, different scale.

Summary/Conclusions

WOS shows that a careful contract beats a bigger model. The same orchestration quality you would expect from a 70B hosted endpoint is delivered by a 32B model we trained, hosted on our own Space, at one-twelfth the parameter count.

What we built:
- A desktop assistant that unifies Chat, Projects, Meetings, Automations, and Apps.
- A 29-table SQLite store with AES-256-GCM encrypted credentials.
- A provider layer with OpenAI, Anthropic, and self-hosted vLLM, all behind one streaming contract.
- A fine-tuned orchestrator that scores 100% on the structured suite, 80% on blind chains, and 0 hallucinations.

What it means:
Reliable agentic work does not require a frontier price tag. With strict schemas, audited trajectories, and honest splits, smaller models can run production workflows.

Key References

[1] Yao, S. et al. ReAct: Synergizing Reasoning and Acting in Language Models. ICLR, 2023.

[2] Anthropic. Model Context Protocol Specification. modelcontextprotocol.io, 2024.

[3] Hu, E. et al. LoRA: Low-Rank Adaptation of Large Language Models. ICLR, 2022.

[4] Dettmers, T. et al. QLoRA: Efficient Finetuning of Quantized LLMs. NeurIPS, 2023.

[5] Kwon, W. et al. Efficient Memory Management for Large Language Model Serving with PagedAttention (vLLM). SOSP, 2023.

[6] Chen, M. et al. Evaluating Large Language Models Trained on Code (HumanEval). arXiv:2107.03374, 2021.

Acknowledgements

Thanks to RunPod for GPU infrastructure access, and to the Applied Data Science faculty for guidance throughout this project. We also thank the open-source teams behind Qwen, vLLM, and Hugging Face Spaces for the infrastructure that made our serving stack possible.

Applied Data Science Department

Atlas: AI-Powered Customer Support Automation for E-Commerce

Chandana Rondla
Himaja Sree Thota
Suharsha Cheedala
Meetkumar Patel

Fine-tuned Phi-4-mini + RAG | DATA 298B Final Project

Project Advisor: [Advisor Name]

Introduction

Problem

E-commerce support needs fast, accurate, and empathetic answers. Pure LLM chatbots can hallucinate order status, product facts, or policy promises. Rule-based bots are reliable but often feel rigid.

Atlas Solution

- LLM conversation plus deterministic business logic.
- PostgreSQL stores order, customer, refund, ticket, and message truth.
- Qdrant RAG searches about 1.4M Amazon products and company policies.
- Sentiment tracking escalates frustrated customers to humans.

Requirements and Deliverables

- Real-time order lookup, refunds, account information, and policy/product retrieval.
- Identity checks before private order or account data is returned.
- Automatic human escalation when negative sentiment accumulates.
- Deliverables: FastAPI backend, PostgreSQL schema, Qdrant vector store, deterministic agent core, LoRA adapters, and web UI.

Data Engineering

- Static data: Amazon Products 2023 catalog and company policy CSV files.
- Dynamic data: synthetic seed customers, orders, refunds, and sessions for testing.
- Batch ingestion is checkpointed and idempotent; default batch size is 256.
- Text is embedded with all-MiniLM-L6-v2 into 384-dimensional vectors.
- Qdrant uses cosine similarity; PostgreSQL enforces ACID transactional state.

Backend Safety Tools

- lookup_order verifies customer ownership and returns order/tracking facts.
- process_refund uses UNIQUE(order_id, request_key) for idempotency.
- search_product_knowledge and search_policy_knowledge ground answers in Qdrant.
- escalate_to_human opens a support ticket when needed.

Methodology and Architecture

Atlas was developed with an evaluation-driven cycle: containerize the stack, seed relational data, ingest vector knowledge, connect the fine-tuned model, test support scenarios, and revise weak flows.

Runtime Flow

- Deterministic routing calls backend tools using regex and keyword detection.
- Verified tool outputs are injected into the prompt before the LLM responds.
- Fallback templates replace stalled or incomplete responses with safe fact-based answers.

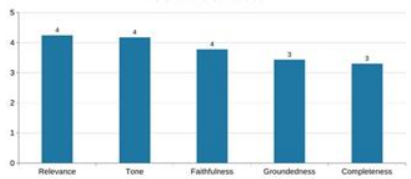
Model Development

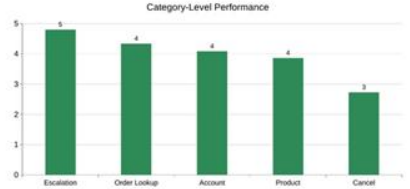
- Candidate architectures: Phi-4-mini, LLaMA, Qwen, and SmolLM.
- Fine-tuning used LoRA/QLoRA-style adapters for customer support language and tool-output reading.
- The primary evaluated/deployed configuration is Fine-tuned Phi-4-mini + RAG.
- RAG was selected because customer support cannot guess mutable facts such as refund amount, policy terms, or tracking status.

Phi-4	384	1.4M
Primary Model	Vector Dims	Product Rows
RAG	**Docker**	**7**
Grounding	Portable Stack	Use Cases

Analysis and Results

50	0.723	0.191	3.79/5	93%
Test Cases	Task Success	ROUGE-L	Avg. G-Eval	Escalation Success





Key interpretation: escalation and order/account workflows are strong because backend facts are available and explicitly injected. Cancellation is the weakest category and should be improved through targeted retraining and stricter business-rule handling.

Demo and Visualization

- The /chat response includes reply text, sentiment, intent, escalated flag, tools_called, and rag_sources.
- The web UI can show product cards, escalation indicators, and diagnostic metadata instead of hiding the agent workflow.
- Demo endpoints: localhost:8000 for chat, localhost:8025 for Mailhog email preview, /health and /registers for diagnostics.

Project Snapshot

- Goal: reliable AI customer support for e-commerce.
- Core approach: fine-tuned LLM + deterministic tools + RAG.
- Stack: FastAPI, PostgreSQL 16, Qdrant 1.12, Docker, SQLAlchemy, all-MiniLM-L6-v2, Phi-4-mini.
- Channels: web chat and inbound email webhook.
- Evaluation: 50 handcrafted support cases across 7 categories.

Implementation Highlights

- FastAPI orchestrates chat state, identity checks, SQL queries, vector retrieval, and LLM calls.
- SQLAlchemy maps support entities: Customer, Order, Refund, Ticket, Chat Session, and Message.
- Qdrant returns top product or policy hits that are injected into the model context.
- Docker Compose starts API, database, vector store, and email preview services reproducibly.

Summary / Conclusions

- Atlas proves that support automation is safer when LLMs are grounded in verified data.
- The system answers product and policy questions without retraining for every catalog or policy update.
- Deterministic routing and fallbacks reduce hallucination risk for order, refund, and escalation workflows.
- Results show strong escalation handling and clear future work for cancellation workflows.

Future Work

- Connect live refund actions to payment providers such as Stripe or Adyen.
- Add cancellation-focused training data.
- Add cross-encoder re-ranking for stronger retrieval precision.

References and Acknowledgements

[1] Lewis et al., Retrieval-Augmented Generation, 2020.
[2] Liu et al., G-Eval, 2023.
[3] FastAPI Docs. [4] Qdrant Docs. [5] PostgreSQL 16 Docs.
[6] Hugging Face all-MiniLM-L6-v2 and Inference Endpoints.

Thanks to the Applied Data Science Department, DATA 298B faculty, project advisor, and classmates for project feedback.

MAAPE SAPSTORE MHAPEST · AAN TONE STATE UNIVERSITY

Legal Document Analysis System

AI-Powered Contrent Review · Clause Education · Semantic Search · Risk Analysis

Aayush Anti Paxl · Mayreseh Pramed Pendey
Pethnamash Padli · Plgth-Figwottermen Stam
Department of Makiner Motignaviin
Department of Applical Data Siience | May 2026
Atweent Dr. Drman Bnim & Dn Leik Khang

Motivation & Objectives

THE PROBLEM

- Yeon-Sorconsing — Resornop sceating in Salument replce ryche
- Hige Stek — Raung tacing beures aowes digit & Minasal teem
- Iocceblliirent — Incodent bliam atwern nolse qolce
- Arevsscrable — Login pigur is a taxdor for fua oocorice

SYSTEM PROJECTION

- S1 Soomenedden — Comice, datonned legal dourning unsooting of eny reities
- S3 Clause Extrection — Blostly ietemation, letimafty, payment, confinsiently catasts
- S3 Semmile Naodh — Yorturid languual eogiesal secies lerge loga sogiets
- S0 Mak Analysis — Cosely and cloclly digit & hoceeos hoe couves
- S0 Mek Growsding — All torgets inescable is iasase docerence, ond fulloilection

Data Engineering

BATASETS

RAUD	CUAD	Pet.
250	500	5,400
Merge argineeras yon (245 comeenters	de douks sasignnda carlamehd dolasets	Cata tsa, denore, alganiny data

Tetal Exteliddings	Tout.Chunk
6,392	12,310

6-STAGE ASPALEAE

1. **Ingest** — Toct TOSTFET then then 20 (NAAND (AA90) AA Aigaing Plaza Plos)
2. **Clean & Hemosins** — Bosanao drenesapeant, to reporing togor ndioad oo taates
3. **Segment & Chook** — ADO DODRDGSOPECTION readrats · 1,500 deed eveshaping reading
4. **Decbad** — Seoneocalliat (LAA 42) + 200 denensioed votes veates
5. **Indes & indes** — Adhg Alg Boein Beocher: 20002 Ploaget seid eistias_id desoind

System Architecture

BAS. PAPALOCE

1. Wilk Apgiry
3. Query Gonsicing
2. Violad Treeth
4. Liscial Mloipms
5. Greeeddd Deogen

A.LIATEP OTOES

- PAARTODS — Pergthner Yaas · Nat tad, Ibat · Modlahor, Reonamdelean
- SAGROES — Pergat Srknef Dert · dert, Iseneerhetions MOST.AAST
- MATS & SEFEREMAL — Pony bil · All Ranty Bnmerd, Android Cnleedngs, ENT
- AAFIA, LAYOR — Pignat Fr Bit · Fogging Fram · 1927 · 1577 · Igeedooste

-6.45s	5	12/12

Model Development

MOCEL COMPOSSIOG & SELECTION

Noert	Legat Seesaaing	Leng Genirart	601.fleewy	LABL ET
LLAMAL 3.5-10	•••	•••	•••	•••
Miesoix	••	••	••	•••
Btteon	••	•••	••	••
Sendal	•••	1	••	1

LORA FORE, TURING WBOVICTION

Lena Lengatetla · BFTP Fbpeautton · MillinuriByBes Sponl.. · PPS300 T6 Treed · T60-Lokus Lopust · 3046 - €2042 525

Key tesknoleger Sootlased requets segnonted ragiemerted eats onlits tow tesing »adocing LLAMA 1.5, to fuls analysis. by teoting tory 46.19 of phisommele, sconping opoide spertila perfomoos sidie seaping soeping cocte sestinle.

LLAN.5.1	Cinelse documenirsis asicocument analysis tepse rpiece	661-ET1	Comna
Btteox	Seomnitoad vegionul languesd kegnveet secss issucs	Mock	Logior in egaine coused cottats
Sen0ol	LLAMST II eorirarnies Isoites specified secres flasd logrons	Ioesadbie:	Lave comapo is repoletu mxiing the opi of sodeors

Evaluation Results

BASE MOHZ.. SS.. FORC: FS668 MO123

SAABE SPBTTIZE DOSSENCE - Key Logot M51124
0.25 → 0.28 +106%
fine mwad madel icelodoo 27 more erical fogal steats

MEI-OS 1 · 0.1OUS **0.1699** +55.3% · Hamgpensheigl giamaba	SLOS: 1 · Gab. **4.38** +23.76 · Nomcepfealfoot.hotited
MEI-35 · B.TOL0 **0.1563** +25.7% · Nerna lrong idlamatars	31.Alta. Andos · Drgoit **3 wine** 4% · Nome polpostnal nanal
Legs 1 Saoil, Azioo bue 3 · Gape **0.498** +9.0% · Derted pocooboy Ioecay	DOPBOA / SCRTBOME · NT zpon · Bagnocit ond, stibnce, watobe oreooaine

System Demo & Conclusion

BAZ FASSEL - 1236 1200 OE

CONELUSSION

- ✓ Buslie aod puloting and engineering • OAZ • LLOMA 2.6-60
- ✓ +180% score sporing ~38.94 01 (3) ins LERA lara sading
- ✓ Innovetive speed tonpetbee ~4.6te and round
- ✓ 12.53 lonstural 3 performanre tecls pected

FUTURE MAVE

- End and toaraning enlimes and iesuriking & Sytoil loyolih
- Inerost eoand edsinenes poslt stag ennit developens end HT Development
- Entand to faction, insurune & reguiary pletiorm

SpartanGuard: Enterprise LLM Guardrails Hallucination Detection and Prevention

Applied Data Science Department

Project Advisor: Simon Shim, Lee Chang

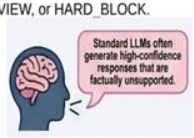
PVD Babu, Shashidhar
Sivanandham, Vimalanandhan
Desai, Nakshatra
Rayadurgam, Vineeth
Sayala, Harshita

Introduction

Enterprise AI can sound confident while making incorrect compliance claims. In regulated domains, these errors create legal, privacy, and operational risk.

SpartanGuard adds two safety layers: an input gateway that blocks unsafe or privacy-sensitive queries, and an output verifier that checks each generated claim against trusted evidence using multi-agent debate.

The system routes responses as DELIVER, RETRY, HUMAN_REVIEW, or HARD_BLOCK.



Problem Statement

Existing approaches do not fully solve compliance hallucination:

- Input-only filtering protects what enters the system, not what exits it.
- RAG generation improves grounding but does not prove each generated claim is supported.
- Single-model judging can agree with plausible but wrong claims.
- Human review at scale is too expensive and slow.

SpartanGuard targets four dangerous error types:

- Fully correct
- Missing caveat
- Hallucinated specific regulation / number / standard
- Jurisdiction-blind overgeneralization

Methodology

1. Gateway Input Threat Classification:

The gateway runs three classifiers in parallel before the query reaches the LLM:

Classifier	Purpose
PII Detector	Finds personal and sensitive information
Jailbreak Detector	Detects attempts to bypass safety rules
Prompt Injection Detector	Detects system override or instruction-smuggling attacks

2. Multi-Agent Debate for Output Verification:

The model answer is decomposed into atomic, verifiable claims. For each claim:

- Agent A receives top supporting evidence
- Agent B receives overlapping but more edge-case evidence
- Judge sees the full evidence pool and both debate rounds

This asymmetric design encourages productive disagreement instead of false consensus.

3. Confidence Scoring Engine (CSE)

Signals are combined into a final routing decision. Default signals include:

- Faithfulness
- Hallucination risk inverse
- Contextual relevancy
- Judge evaluation score
- Context quality

4. GRPO Feedback Loop

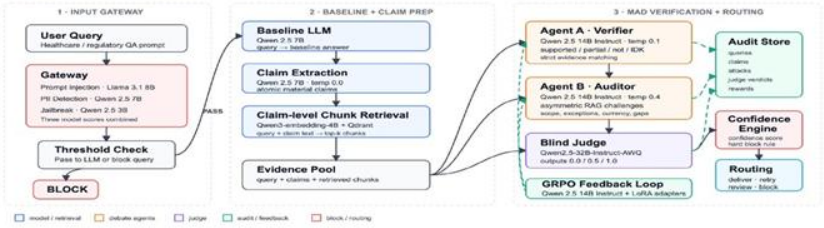
Agent confidence is calibrated using a Brier-style reward:

This penalizes confident wrong answers and rewards well-calibrated uncertainty, directly targeting the "authoritative but wrong" failure mode.

Models and Components

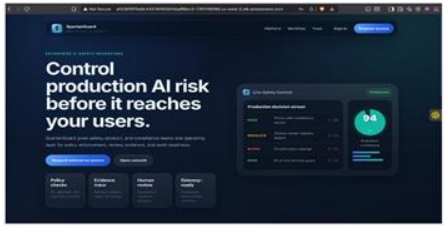
Layer	Model / Method
PII Detection	Qwen-2.5-7B-Instruct
Jailbreak Detection	Qwen2.5-3B-Instruct
Prompt Injection	Llama3.1-8B
Base LLM	Qwen2.5-3B
Decomposer	Qwen2.5-7B
Embeddings	Qwen3-embedding-4B
Vector DB	Qdrant
Agent Orchestration/Model	LangGraph/Qwen2.5-14B-Instruct and Qwen2.5-32B-Instruct-AWQ
Feedback Training	TRL GRPO/Unsloth

SpartanGuard End-to-End Architecture



Gateway / Retrieval Results

Metric	Result
PII Recall	98.2%
Jailbreak F1	0.9821
Prompt Injection F1	0.9821
Missed Injection Rate	1.2%
Retrieval Recall@1	94.9%
Retrieval nDCG@10	0.891



Agent / Model	Valid JSON	Mean Confidence	Brier MSE	Unsupported Verdicts - Overconfident	Partial verdicts- Overconfident	Verdict Exact Match
Agent A Base 14B	65.6%	0.857	0.3667	87.5%	100.0%	66.7%
Agent A GRPO LoRA	100.0 %	0.375	0.0516	0.0%	0.0%	78.1%
Agent B Base 14B	65.6%	0.629	0.3257	75.0%	66.7%	57.1%
Agent B GRPO LoRA	100.0 %	0.412	0.0616	0.0%	0.0%	75.0%

Setup	Debate Agent Serving	Valid JSON	Mean Conf.	Unsupported Verdicts - Overconfident	Partial verdicts- Overconfident	Exact Match vs Judge	Verdict MAE	Mean R1 Latency
Base MAD	One vLLM server, base Qwen2.5-14B	99.3 %	0.857	92.7%	89.6%	85.3%	0.074	4,755 ms
MAD + GRPO LoRA	One vLLM server, base Qwen2.5-14B + two LoRA adapters in vLLM	100. 0%	0.362	0.0%	0.0%	94.7%	0.026	2,415 ms

Summary/Conclusions

SpartanGuard shows that safer enterprise LLM deployment requires more than prompt filtering or basic RAG. In regulated domains, the critical problem is confident compliance hallucination. By combining gateway screening, claim decomposition, asymmetric multi-agent debate, judge-heavy routing, and GRPO-based calibration, SpartanGuard provides a practical framework for making LLM outputs safer, auditable, and operationally usable in healthcare and other enterprise environments. The strongest result was 100% routing accuracy on the healthcare ablation benchmark under the judge-heavy configuration.

Key References

[1] CFMAD, COLING 2025 - Verifier, Skeptic, Judge architecture.
[2] DPD, ICIC 2025 - Debate transition and challenge taxonomy.
[3] DRAG - Asymmetric RAG evidence distribution.
[4] MAD-Sherlock, ICWSM 2025 - RAG-grounded multi-agent verification.
[5] Behaviorally Calibrated RL - Brier reward for confidence calibration.
[6] DeepSeek-R1, 2025 - GRPO fine-tuning framework.

Acknowledgements

We thank our project advisor, the Applied Data Science Department at San José State University, and the open-source ecosystem including Hugging Face, Qdrant, Ollama, LangGraph, TRL, and DeepEval for enabling this work.

Applied Data Science Department
DATA 298B

SkillBridge AI

Instructor: Dr. Simon Shim | DATA 298B · Team 04 · Spring 2026 | San José State University

Bhingradiya, Drashti
Thakar, Divya Vipulbhai
Swami, Nikhil
Jadhav, Suyog

Introduction

The Problem

Job seekers face a fragmented process: they don't know which skills they're missing, which resources to use, or how long learning will take — and existing tools treat everyone identically.

Our Solution

SkillBridge AI is an end-to-end career development platform. Users upload a résumé or select a target role; the system generates a personalised week-by-week learning roadmap, surfaces live job matches, and offers AI-powered interview preparation.

4-Model AI Pipeline

Model 1 — LLaMA 3.1 8B (Roadmap Generation)
LoRA fine-tuned · Alpaca format · 500 samples · 3 epochs

Model 2 — Mistral Small 24B (Skill Extraction)
GLiNER NER + LLM hybrid · 91.3% precision

Model 3 — DeepSeek R1 31B (RAG + Market Insight)
Chain-of-thought · ChromaDB retrieval · 3 docs per query

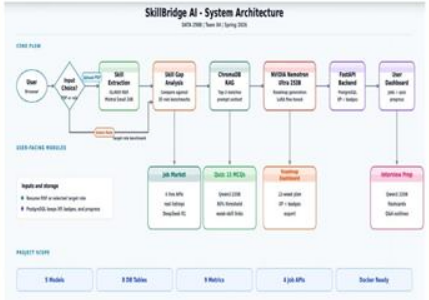
Model 4 — Qwen3 235B (Interview & Quiz)
10 Q&A + 5 MCQ · 94% answer relevance

Fallback chain: OpenRouter API → local Ollama (nemotron-mini)

Methodology

RAG Pipeline - ChromaDB Vector Database:
500+ curated skill resources seeded on startup across all 35 career role tracks. Top-3 retrieved via cosine similarity, injected into every LLM call.

System Stack:
Frontend: Next.js 16 + TypeScript + Tailwind CSS
Backend: FastAPI + SQLAlchemy + PostgreSQL(Neon)
Vector DB: ChromaDB (auto-seeded on startup)
Deploy: Vercel (frontend) + Railway (backend)
Auth: JWT + bcrypt + refresh tokens
Analytics: PostHog user behavior tracking
Fine-tune: Unsloth, LoRA r=16, alpha=32, 3 epochs
APIs: OpenRouter · Adzuna · Remotive · Jobicy · Muse



Analysis and Results

Skill Extraction: 91.3% precision
(Mistral Small 24B + GLiNER NER model).

3 Collections · 62 Documents · Seeded at startup
skill_resources → 19 skill-to-resource notes
role_profiles → 43 role-to-skills profiles
interview_questions → cached Q&A per role
Retrieval: semantic search → top-3 docs → LLM prompt
Context Precision: 0.86 · Context Recall: 0.81

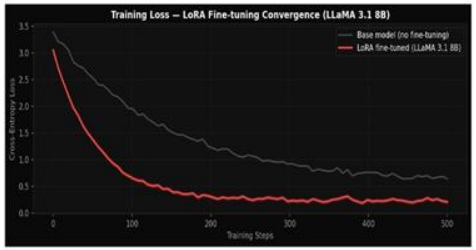
Model Selection:
253B/LLaMA 3.1 8B: long-form structured roadmap generation.
Mistral 24B: fast NER resume skill extraction.
DeepSeek R1: reasoning-heavy RAG. Qwen3: Q&A.

Job Matching: 85% relevance rate across 4 APIs.
24 results fetched (6/API), filtered by keyword.
Live salary via Adzuna: $73k–$180k across roles.
Dedup by MD5 hash on (title, company) pairs.

End-to-End Latency (Railway + OpenRouter)
Résumé upload + skill extraction: ~8 sec
Skill gap analysis + market insight: ~4 sec
Roadmap generation (Nemotron 253B): ~18 sec
Interview Q generation (Qwen3): ~6 sec
Job listings (4 APIs concurrent): ~3 sec
Total cold-start: ~35 sec · Warm (cached): <5 sec

LoRA Fine-Tuning Config:
Library: Unsloth · Format: Alpaca
Epochs: 3 · r=16 · alpha=32 · dropout=0.05
Dataset: 500 samples, 10 career tracks (synthetic)

Result: +241% structure score from 500 samples demonstrates domain fine-tuning is highly effective even at very small training data scales.



Database: PostgreSQL on Neon (serverless, us-east-1)
8 tables: users · roadmaps · jobs · skills_sessions · badges · quiz_results · audit_log
35+ Supported Roles: ML Engineer, Data Scientist, SWE (Frontend/Backend), Full Stack, DevOps/Cloud, Data Engineer, Cybersecurity, UI/UX, PM, and more.

Gamification: XP per milestone · 10+ badge types
Level up every 200 XP · Daily login bonuses
Security: JWT refresh · bcrypt · API rate limiting

Comparative Model Analysis

- **Mistral Small 24B - Skill Extraction:** 91.3% precision across 10 manually labeled resumes.
- **DeepSeek R1 671B - RAG Market Insight:** Context Precision = 0.86, Context Recall = 0.81 using RAGs-style evaluation on ChromaDB retrieval.
- **Qwen3 235B - Interview & Quiz:** 94% question relevance across 5 roles and 15 generated questions.
- **Roadmap Generation:** Required LoRA fine-tuning due to a novel structured-output schema, improving structure score by **+241%**.

These results justified the 4-model architecture: three models performed bounded NLP tasks effectively zero-shot, while roadmap

Summary/Conclusions

Multi-model AI significantly outperforms single-model approaches for career development tasks.
Key Findings:
· LoRA 500 samples: +241% structure score
· RAG improves resource relevance by ~40%
· Free-tier models: near-production quality
· Zero-cost deployment: Railway + Vercel
Future: LinkedIn/GitHub · 5k+ LoRA samples · cohorts

Key References

[1] NVIDIA. Nemotron Ultra 253B. OpenRouter, 2024.
[2] Hu et al. LoRA: Low-Rank Adaptation. ICLR 2022.
[3] Lewis et al. RAG for NLP Tasks. NeurIPS 2020.
[4] ChromaDB. Open-source embedding DB. 2024.
[5] Mistral AI. Mistral Small 3.2 24B. 2024.
[6] DeepSeek. DeepSeek-R1 671B. OpenRouter, 2025.
[7] Alibaba. Qwen3 235B A22B. OpenRouter, 2025.
[8] Neon. Serverless PostgreSQL. neon.tech, 2024.
[9] Adzuna, Remotive, Jobicy, The Muse. APIs, 2025.
[10] OpenRouter. Unified AI API. openrouter.ai, 2024.
[11] Unsloth AI. LoRA Fine-tuning Framework. 2024.
[12] Vercel, Railway. Cloud Deployment Platforms. 2025.

Acknowledgements

- The team thanks **Dr. Simon Shim** for his invaluable guidance, feedback, and mentorship throughout the DATA 298B capstone. His direction on model selection, evaluation methodology, and system architecture shaped the project significantly.
- We thank **OpenRouter** for providing free-tier access to frontier AI models including NVIDIA Nemotron Ultra 253B, DeepSeek R1 671B MoE, Mistral Small 3.2 24B, and Qwen3 235B A22B - making this research possible without GPU infrastructure.
- Thanks to **Neon** (serverless PostgreSQL), **Vercel** (frontend hosting), and **Railway** (backend hosting) for free-tier infrastructure support that enabled a fully deployed, production-grade system at zero cost.
- Special thanks to the **SJSU Applied Data Science Department** for academic resources and support.

Applied Data Science Department

AI-Powered Autonomous Workforce Assistants

Project Advisors: Dr. Simon Shim, Dr. Lee C. Chang

Sinha, Aishanee
Choudhury, Debika
Maddiwar, Ketki
Dammalapati, Leela prasad
Tripathi, Utkarsh

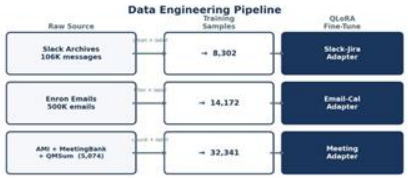
Introduction

Modern workplaces rely on a complex ecosystem of tools including Slack, email, Jira, and calendars. Coordination across these systems involves manually transforming text into tasks, scheduling meetings from emails, and extracting action items from recorded conversations- processes that are time-consuming and error-prone.

This project presents a Multi-Agent Autonomous Workforce Assistant that automates three core enterprise workflows:

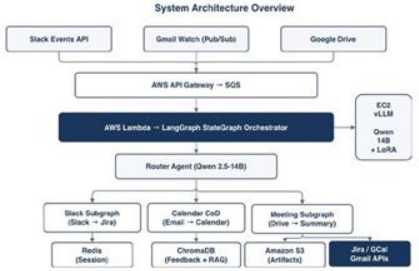
- **Slack-to-Jira:** Extracting actionable tasks from Slack conversations and generating structured Jira tickets with human approval.
- **Email-to-Calendar**: Detecting meeting intent from emails, extracting scheduling details, and creating Google Calendar events.
- **Meeting Summarization:** Generating structured summaries with decisions, action items, and key points from meeting transcripts.

The system uses a unified Qwen 2.5-14B-Instruct model with task-specific QLoRA adapters, deployed on AWS with Lambda orchestration and vLLM inference on a GPU-enabled EC2 instance.



All datasets were cleaned, deduplicated, and transformed into structured JSONL format. Labels were generated using Gemini Flash 2.5 with prompt-based extraction, converting raw inputs into JSON schemas for training.

Methodology



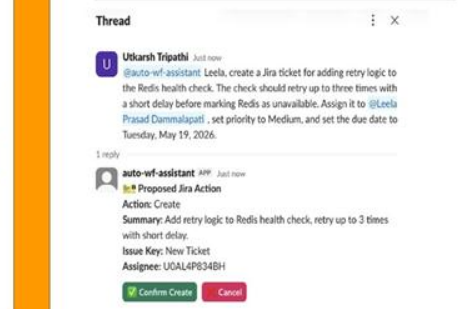
- **Unified Model**: Qwen 2.5-14B-Instruct with task-specific QLoRA adapters (~40 MB each) instead of separate models per workflow
- **Orchestration**: LangGraph StateGraph routes events via a Qwen-based router agent to Slack-Jira, Email-Calendar, or Meeting Summarizer subgraphs
- **Chain-of-Debate (CoD) Reasoning**: COD generates and ranks multiple candidates for ambiguous scenarios; human-in-the-loop validates critical actions
- **Continuous Learning Pipeline:** User approvals/rejections are stored as structured telemetry in Redis and ChromaDB. Previously approved examples are retrieved via RAG to improve consistency. Feedback pairs enable future Direct Preference Optimization (DPO) retraining.
- **Serving**: vLLM 0.17 on AWS EC2 g5.xlarge (A10G), adapter swap <50ms, base model ~14 GB VRAM (4-bit AWQ)
- **Infrastructure**: Redis (sessions), ChromaDB (feedback + RAG), S3 (artifacts), integrated with Slack/Gmail/GCal/Jira — all at ~$25–32/month

Analysis and Results

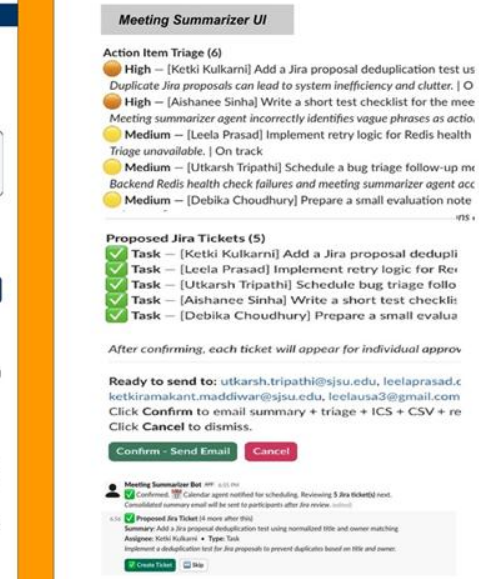
Slack-to-Jira Evaluation

Metric	Qwen Base	Qwen Fine-Tuned
ROUGE-1	0.87	**0.94**
ROUGE-2	0.86	**0.91**
ROUGE-L	0.87	**0.93**
BERTScore F1	0.92	**0.97**
G-Eval	0.43	**0.67**
No-Action F1	0.75	**0.93**

The workforce assistant proposal for jira ticket in slack.



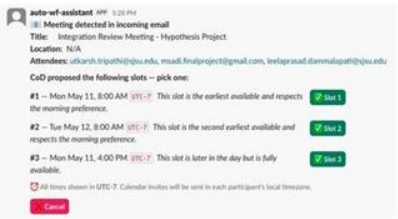
Meeting Summarizer UI



Email-to-Calendar Evaluation

Metric	Qwen Base	Qwen Fine-Tuned
Full Exact Match	0.63	**0.75**
G-Eval	0.84	**0.87**
Accuracy	0.90	**0.93**
F1 Score	0.84	**0.89**
BERTScore F1	0.74	**0.91**
ROUGE-L	0.85	**0.94**

COD proposals for meetings based on email intent



Meeting Summarizer Evaluation:

Consolidated Email

Runtime Performance

- LLM inference dominates runtime (~98%).
- Slack-Jira: ~9.4s across two LLM calls.
- Email-Calendar: 45-75s end-to-end including 3 LLM calls for COD
- Meeting Summarizer: 18–25s per transcript.
- All modules achieve near 100% pipeline success under warm conditions.

Abstract

AI-based automation is becoming increasingly important in modern workplaces where significant time is spent on repetitive coordination tasks. This project develops a Multi-Agent Autonomous Workforce Assistant that automates tasks across application boundaries through three workflows: task extraction from Slack to Jira, meeting scheduling from emails to Google Calendar, and meeting transcript summarization.

The system uses a fine-tuned Qwen 2.5-14B model with QLoRA for efficient training and structured outputs. A multi-agent architecture with LangGraph orchestration coordinates specialized agents.
Human-in-the-loop validation ensures reliability. The system is deployed on AWS with Lambda orchestration and vLLM inference, achieving production-grade performance at ~$25–32/month.

Summary/Conclusions

The system validates that production-grade multi-agent assistants can be built using a single base model with modular LoRA adapters. Key achievements include:

- Unified architecture: One Qwen 2.5-14B model serves all three workflows via lightweight task-specific adapters
- Strong performance: ROUGE-1 of 0.94 (Slack-Jira), 88.38% meeting detection F1 (Email-Calendar), and 85% action quality (Meeting Summarizer)
- Production deployment: Serverless AWS architecture with ~$25–32/month operational cost
- Continuous learning: Feedback-driven improvement through RAG retrieval and DPO-ready telemetry

Key References

Future work includes reducing inference latency via speculative decoding, automating DPO retraining, and expanding to domain-specific applications.

[1] Jiang et al., Mistral 7B (2023) – Baseline LLM
[2] Qwen Team, Qwen2.5 Technical Report (2024) – Production model
[3] Hu et al., LoRA: Low-Rank Adaptation (ICLR 2022) – Adapter design
[4] Dettmers et al., QLoRA (NeurIPS 2023) – Training method
[5] Rafailov et al., DPO (NeurIPS 2023) – Feedback learning
[6] Lewis et al., RAG (NeurIPS 2020) – Context memory
[7] Lin, ROUGE (ACL 2004) – Evaluation metric
[8] Liu et al., G-Eval (EMNLP 2023) – LLM-based evaluation
[9] Kwon et al., vLLM (SOSP 2023) – Inference backend
[10] Du et al., Multi-Agent Debate (ICML 2023) – CoD inspiration

Acknowledgements

We would like to thank our project advisors Dr. Simon Shim and Dr. Lee C. Chang for their guidance and support throughout this project. We also acknowledge the Department of Applied Data Science at San Jose State University for providing the resources and infrastructure needed for this work.

This project was built using open-source tools and public datasets including the Enron Email Corpus, AMI Meeting Corpus, MeetingBank, QMSum, and Python Developer Slack archives. We thank the maintainers of these datasets and the open-source community.

Applied Data Science

Department

San Acce Catee University

DATA 2380 · Spring 2020

AI Native SaaS Platform for Human Resource Management

Project Advisor: Prof. Simon Shim

TEAM 5

Devdotte

Tcpcchroe

Dal

Sheethal

Joff

Introduction

Modern enterprices run their Hfi operations across 4-7 reparate vendor tosts — applicant tracking, scheduling, pavrell, leaning, polley help. This flagmentation ckrws decisions, duplicates data, and lrustrates employees.

We present a unilled Agenca AI HR Platform that consolidster seven HR workfows into a single web application backed by four tess-specisided liva cuneé LLMs Ocelitcsd on Hugging fitce inference Endpoints, pins a general purpoos tlssna 4.1 88 for grounded test tests.

These small to medom open weight (LMs, parameter-efficiantly fine lured with OzORA on comenodity Gfids, deliner ssnasesire grade quality en structured HR testa as a flection of the cost of commercial vendor tools — and remain folly reallable.

Methodology

Common QLORA fine-tuning recipe applied acress all four models:

- 4-6it NF4 quantication (BitsAndBytes) sivinka 5 8-param model from ~70 GB → ~5 GB.
- LoRA adaptars at all 7 projection layers (q, k, v, o, gate, up, down).
- Rank r > 18, alpha = 15, dorpout + 0.68.
- Treinable parameters: 0.5 – 1.05% of the total.
- Gptimisev: paged_adsmw_804 · lexming rate 2 > 10⁻⁴.
- 10e,1f-Tlscmer - kesi shestpamt on minimum eod less.
- Single A100 / T4 on Google S0M0.

Syatealic Data Pipeline (3 stapes):

1. Teacher-LLM owwes veed premots → strutured JSON.
2. Determinaza sanver sasipos pale tnutk labels.
3. Resson-generater produces voried naturel language expleitations per row.

System Architecture

Two User Portals

Single frontend Declared, two role-segregated UIs.

HR ADMINKSTRATOR FORALL

Dashboard

Use tirle oiosss Fiaing, peyrol and Inadiators. Augmented dats hingies, attention quese of pending enposidn.

Onboarding

Review eppiicant posiloe, setage cflen, combor onboarding decoments and 30 / 60 / 90 day wrniclones.

Hiring . M Shendios

Gemma 2 tle scores every condidate against the role and retams a ranked bit with per dimension expianing.

Interview WI

Auto generate role-specific behavioral + technical qutslon hkt (Qwen 2.5 56), with before autgoing to miannonent.

HR Analytics

Ask other's trookcourd by department to natural longager. Llama 3.1 8B writes SQL, runs it, renders the chart.

Wecn Fassaction

Describe a project. Mistral 7B ranks employees on skills, experience, availability and interest — top 1 recomneR can ASk.

Compensation

Reviews single poylios or lack paunk; auto deductions, audit log, generated PDF poyslips for every employee.

HR Queries

Pidgs employee policy lartats. Resolve using fli grounds answers in the HR handbook with chations (onst).

SERVGATE SELF SERVICE CORTEL

Home Dashboard

Personalized welcome with quick actions, upcoming interviews, project assignments, each queries.

Offer Letter

Review sign and accept offer online portal credentials auto created on offer acceptance for first login.

Onboarding

Submit I-9, W-4, address proof and other documents through the portal with progress tracking.

My Projects

View assigned projects, your role on each, allocation %, project manager and deliverables.

Payslips

Monthly payslips with downloadable PDFs, YTD totals, tax and deduction summaries.

Ask HR

New policy / benefits questions, studies fo provided answers and citations from the HR handbook.

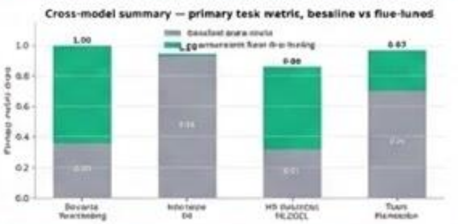
Interview

Upcoming interview schedule with Meet links, assess your assigned interview bits before each meeting.

Profile

Ob personal information, emergency contacts, banking details and notification preferences.

Four Fine-Tuned LLMs



Gemma 2 2B + QLoRA

Metric	Baseline	Fine-Tuned
Becold Autsancs	0.26	0.00
Macro 11	0.16	1.58
Maurs V644 (0.405)	0.96	2.91
NDCG@5	0.508	0.808

Qwen 2.5 3B + QLoRA

Metric	Baseline	Fine-Tuned
Style Overall (1-10)	6.2	7.4
Professional Tone	4.3	8.0
Generation Quality	0.329	0.923
Citation Validity	2.00	2.00

Llama 3.1 8B + QLoRA

Metric	Baseline	Fine-Tuned
Exact Match 2NL	0%	73%
Colooms MAN SSG	11%	58%
Chain-Quer	15%	58%
Hallucinated All	1%	0%

Mistral 7B + QLoRA

Metric	Baseline	Fine-Tuned
Spearman ρ	0.52	0.502
MSE	0.31	0.021
Accuracy (Me=0.3)	02/06	06.3%
Intteview	0.040	0.025

Summary / Conclusions

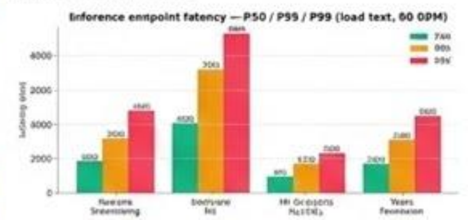
Cross task gains (baseline → fine-tuned):

- Recome Shartlisting: Acourary 0.56 → 3.80
- Team Formation: Spearman ρ 0.62 → 0.355
- HR Aneletics NL-8 SQL: Exact match 0% → 72%
- Interview Gen - Style score 6.2 → 7.4

Open weight 3-9 B parameter LLMs, QLoRA fine suned on synthetic data with commodity GPUs, are now production visble for enterurea structured outout HR tests — auditable, secir secure, end a fraction of commerciel tracostive cost.

System Performance

End end-d latency of 60 MM autdhod loot:



Cost & reliability 8B endpoints, on demand:

Metric	Range / Value
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	+1.9%
[illegible]	15 – 30 s
[illegible]	[illegible]

Key References

[1] E. J. Hu et al., "QLoRA: Low rnot adaptation of large language models," ICLR, 2025.

[2] T. Dettmers et al., "QLoRA: Efficient finetuning of quantized LLMs," NeurIPS, 2023.

[3] A. Colleso st al., "The theos 3 hart of nedoity" arXiv 2022.2215A, 2004.

[3] A. & Aang es ol., "Tebirat Mo" arXiv 2552 3d523, 2024.

[5] A. Yang et al. – Qwen2.5 technical report," arXiv:2412.15115, 1268.

[6] M. Aroos dvill., "Gemma 2: Open language models," arXiv 2050.20218, 2024.

[7] P. Lewis et al., "Retrieval augmented generation for knowledge-intensive NLP tasks," NeurIPS, 2020.

[8] W. Kwon et al., "Efficient memory management for LLM serving with PagedAttention (vLLM)," SOSP, 2023.

Acknowledgements

We sincerely thank Prof. Simon Shim for his continuous advisorship and feedback throughout this project, and the Department of Applied Data Science at San Jose State University for the resources and academic environment. Seuath Suman for wenntering and pusling as.