Automated Data Extraction from Coroner's Reports

Client: Dr. Matthew Albrecht

Group: 18

GitHub Repo: https://github.com/AdamUWA/coroner

1. INTRODUCTION:

1.1. Project Objectives and Data Description:

This project focuses on development of a self-contained system that can securely process Coroner's reports to extract critical insights regarding the causes and implications of unexpected, unnatural, or unexplained fatalities. These reports comprise disparate, unstructured data on factors such as medical conditions, socio-economic factors, accident circumstances, and alcohol or substance presence. The content insights generated by the system are intended to improve the data analysis capabilities of the WA Centre for Road Safety Research (WACRSR) in order to identify and recommend preventive measures against surging road transportation fatalities.

The reports are typically PDF-based but often inconsistently formatted comprising various structures like text, tables, graphs, and images. For instance, one of the source data, Nicholls-Driver-Finding.pdf is a scanned image copy of document while the other source data, Rodier-Finding.pdf is a text-based PDF file. Although the cloud-based solutions may offer easier extraction from these varying data sources, the sensitive nature of the Coroner's report strictly necessitates local execution, prohibiting such cloud-based processing solutions.

The manual review process currently employed by the group of researchers like our client is time intensive and prone to human error. Analysing the dynamics, this system has been built. It can automate data extraction ensuring local execution to address data security concerns. This scalable system offers complete or semi-automated solutions to process and extract information from the documents.

1.2. Summary of Approach and Contribution:

We implemented a modular Retrieval- Augmented Generation (RAG) pipeline that runs entirely on a local machine to fulfill the project's objectives. This architecture leverages the capability of Large Language Models (LLMs) for natural language understanding, while anchoring their outputs in the factual content of the source documents. This grounding mechanism reduces the risk of LLM hallucination.

The pipeline initiates by extracting textual contents from the ingested PDF reports with the application of Optical Character Recognition (OCR). The extracted text is segmented into vectorisable chunk and then projected into a pre-trained embedding space. These embeddings are stored in a locally hosted vector database to ensure adherence with data privacy protocols. When a query is passed by a user, the system retrieves the most contextually relevant text segments. These segments are projected as context to the LLM, which then generates a response.

The system is designed to be modular, allowing for the evaluation of the 3 LLMs used (Llama 3.2, Gemma3, Phi4-mini) to assess the trade-offs between performance and computational cost. The entire workflow is orchestrated locally using OLLAMA for model hosting, ensuring full data privacy and security.

The Contributions include:

i. Modular Python-based system using LangChain for RAG

ii. Ollama for local LLMs and pre-trained embeddings (gemma3. Llama3.2, phi4-mini, mxbai)

iii. BERTScore for evaluation

iv. Pre-vectorized JSONL files for efficiency

v. Textual (terminal), Graphical (local webapp UI), and programmatic (Python script) interfaces

vi. Evaluation results

This yields a scalable system capable of processing reports in minutes detailed in our Github repo for reproducibility.

2. METHODS AND RESULTS:

This section records all the architectural decisions, implementation specifics, and the dual-pronged evaluation methodology employed in this project. Every choice made during the development of the project is inspired by the unique constraints and the objectives of the project.

2.1. Reasons for Selecting the Architectural Approach

Locally deployed Retrieval-Augmented Generation (RAG) pipeline is the core of the project. The approach was chosen over alternatives like regular-expression matching, LLM fine-tuning, or direct querying of a base LLM for various critical reasons grounded in recent NLP research:

i. **Mitigation of Factual Hallucination**: Standard LLMs are very much prone to generating factually incorrect information, known as hallucination. Hallucination is an unacceptable risk for the domain like Coroner's report which requires high accuracy as it deals with sensitive data. The RAG grounds the LLM's response in explicit evidence retrieved from the source document and forces the model to synthesize answers based on provided text over its parametric knowledge. This technique mitigates the chances of hallucination eventually enhancing factual consistency (Lewis, Perez and Piktus).

ii. **Adherence to Security and Privacy Constraints**: Data security and privacy is a non-negotiable requirement of the project due to its sensitive nature. This precluded the use of cloud-based AI services and inspired a self-contained system. The selection of Ollama for local model serving, combined with open-source LLMs and a local in-memory vector database, ensures that sensitive data from the Coroner's reports never leave the user's machine.

iii. **Enhancing Explainability and Auditability**: Keeping the LLM hallucination in mind, an ability to verify the system generated output was a mutually agreed upon requirement. Unlike a standard LLM that acts as a black box making it difficult to trace the evidence of the generated response, the RAG implementation offers source attribution for every response it generates. The framework links all information back to the specific page and text chunk in the source PDF. This significantly bolsters accuracy by not only allowing us to trace the evidence but also helping to re-verify the context and the results (Binwal and Chopra).

iv. **Flexibility and Scalability**: RAG offers more flexible approach, as the knowledge base can be updated simply by adding new documents to the vector store without any model retraining (Gao, Xiong and Jia).

2.2. System Implementation and Experimental Methodology

The methodology is broken down into the distinct phases of the RAG pipeline, followed by a robust, hybrid evaluation framework.

i. **Data Preprocessing (OCR and Chunking):** The Coroner's reports are processed using EasyOCR to extract text via OCR. The documents are chunked into ~500-token segments with metadata (page, source) for vectorization. The chunks with associated metadata are then pre-vectorized into serial object files (JSONL) for efficient vector store loading.

ii. **Vector Embeddings and RAG Integration:** The text chunks are transformed into vector representations using a pre-trained embedding model(mxbai-embed-large) and stored in memory vector database (Langchain). When queries are made, they are also embedded similarly. Then the system retrieves top-k (k=3) similar chunks based on cosine similarity. The retrieved contexts are augmented with queries via a prompt template. This ensures contextually and factually correct generations of results.

iii. **LLM Query System and Response Generation:** Local LLMs generate responses from augmented prompts. Chat Prompt Template with context, query, and instructions for no prior knowledge outputs response + sources.

3. DISCUSSION

3.1. Comparison Between Outputs from Different Tested Methods

All the three LLMs were compared with 10 queries on blood/chemical analysis reports and 15 queries across 6 Coroner's reports with 3 tiers of difficulty with the queries like:

i. **Easy** – Direct factual extraction (e.g. “Who is the deceased?”)

ii. **Medium** - Required synthesis of information (e.g. "What were the circumstances leading to the death?")

iii. **Hard** – Higher order reasoning (e.g. "Summarize the report")

When the outputs scored via BERT F1 for semantic alignment with manual ground-truth answers, were evaluated it was found that the Gemma-3 model was the best-performing and most balanced choice for the project. Although the aggregate F1-score for both Gemma-3 (0.875) and Llama-3.2 (0.862) were statistically close, the breakdown revealed critical differences. While Llama 3.2 performed well on blood/chemical analysis queries, Gemma3 did consistently well when tested on Coroner's report. This suggests that there is not a single that is universally optimal.

The most critical insights emerged from the precision-recall breakdown. Gemma3 demonstrated higher precision (0.86 Vs 0.85), meaning it is more conservative and less likely to introduce factually incorrect information. Whereas Llama3.2 had better recall (0.87 Vs 0.868), capturing a wider range of available information. With the nature of data and analysis the project is intended to, it is better to flag an uncertain field for human review than to populate the database with incorrect information that could corrupt the entire dataset and lead to flawed conclusions. The Llama3.2 had slightly better recall value averaging ~ 0.87, indicating it can capture more of the available information.

Both larger models Gemma-3 and Llama3.2 averaged F1 score greater than 0.85 on easy and medium level while only managed to average 0.78 on hard level. On the other hand, the smallest model Phi-4 Mini with an average F1 score of approximately 0.85 and the average precision of around 0.83, was the fastest model but couldn't compete with the larger models in terms of precision and recall. Gemma3's edge in precision makes the model more suitable for error-adverse tasks like extraction while Llama3.2's consistency makes it a preferrable choice for batch processing.

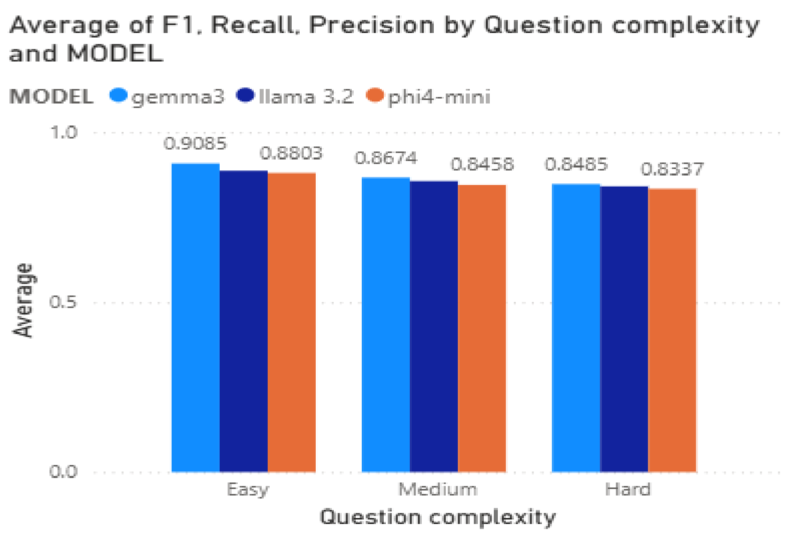
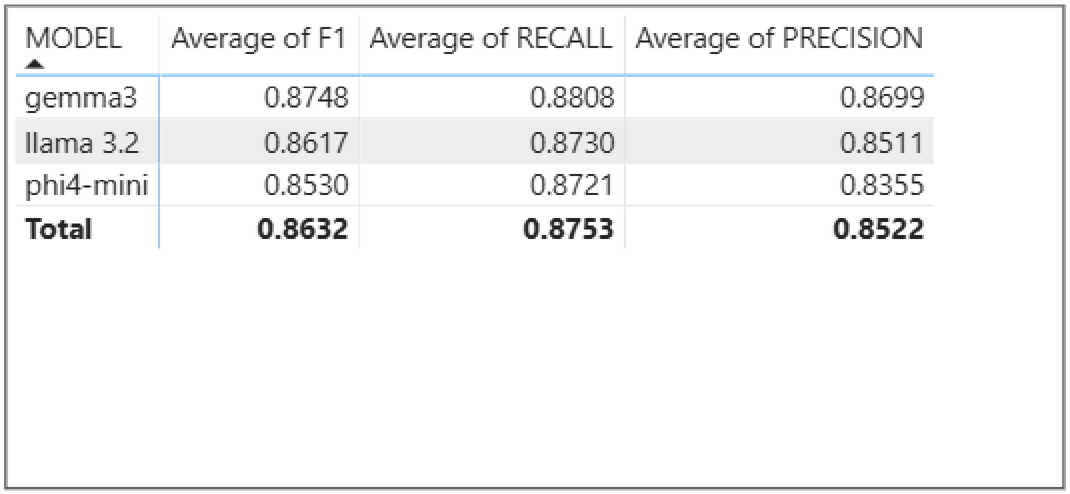


Figure 1: Average of BERT metrics on all reports.



MODEL	Average of F1	Average of RECALL	Average of PRECISION
gemma3	0.8748	0.8808	0.8699
llama 3.2	0.8617	0.8730	0.8511
phi4-mini	0.8530	0.8721	0.8355
Total	**0.8632**	**0.8753**	**0.8522**

Figure 2: Overall Model Performance of all three models across all difficulty levels(Average BERT Scores)

3.2. Strengths and Limitations of the Tested Methods

i. Strengths:

Automation and Consistency: The automated pipeline allows for consistent testing of models without human intervention in the scoring system.

Three-Tier Difficulty Structure: The tiered framework allowed us to diagnose where models struggle, distinguishing between failures in simple retrieval versus complex reasoning.

Local Execution: The methodology successfully demonstrated that the project's strict security requirement is achievable without compromising functionality.

ii. Limitation:

BERTScore vs. Factual Correctness: BERTScore measures semantic similarity, not factual accuracy. A model could generate a response that sounds correct and scores well but contains subtle factual errors.

Retrieval Limitations: The Top-k=3 retrieval setup might have limited model performance on complex questions requiring information from more than three sections of a report. It missed cross-paged context (e.g. medical history on page 2 influencing page 6 cause).

Limited Sample Size: The evaluation was conducted on six publicly available Coroner Report, which may not capture the full diversity of fatality cases.

3.2. Consideration on Explainability, Responsible Use, and Potential Bias

The built-in explainability of RAG is its primary advantage. Each answer is transparently linked to its source chunks including page numbers, and document names. This feature allows researchers to trace the extractions and verify its correctness.

Despite high accuracy, the system shouldn't be fully autonomous, at least in the earlier stage of deployment. To support responsible use, the generated outputs having low-confidence retrievals (<0.7 cosine) are flagged for human review. It is best deployed as a tool to accelerate initial data extraction rather than an autonomous system.

There could be a couple of bias risks. Embedding may undervalue legal-medical jargons, favoring more common terms as it was trained on general web text. Additionally, our ground-truth reflects group interpretations that may overlook cultural nuances in socio-economic factors.

4. CONCLUSION

4.1. Comprehensive Summary

The project successfully demonstrated a fully self-contained document processing pipeline that extracts structured information from sensitive WA coroner reports. The RAG system, designed with privacy at its core, directly addresses WACRSR's needs for scalable road fatality analysis.

By integrating OCR, local embeddings, and grounded LLMs, we were able to transform manual, error-prone reviews into an automated pipeline that achieves ~0.86 accuracy all while ensuring that sensitive data remains on the local machine. The evaluation results confirm the production readiness of both the Gemma3 and Llama3.2 models, with their transparent and trustable results. The modular system hosted on GitHub supports immediate extension, meeting our objectives of efficacy, consistency and security.

4.2. Future Improvements

Although the current system establishes a strong and functional framework, several tactical directions have been identified to anchor the project forward for its performance and expandability. The following areas represent the promising opportunities to advance the research and transition the system to operational-level use at WACRSR:

i. **Domain Specific Model Specialization**: The LLMs currently employed in this model are general purpose model. Though these models offer accuracy of ~86%, the system could benefit by iterating it with the integration of domain specific models, especially with those fine-tuned on formal legal or medico-legal corpus.

ii. **Embedding Model Evaluation**: mxbai-embed-large embedding model, the current embedding model employed in this model is approximately 669 MB in size. Though this model demonstrates adequate performance, it is unclear if the application of larger or smaller embedding model offers comparable results with reduced computational energy. Future work could test and systematically test and compare embedding models of various sizes to find the best balance between efficiency and performance.

iii. **Hardware-Optimized Processing Pipeline**: The current framework emphasizes accessibility and simplicity over higher computing performance. The impact of hardware acceleration like GPU-optimized OCR and preprocessing pipelines could be tested for the performance boost of the system, in the future.

5. REFERENCES

Binwal, Ankur and Puneet Chopra. "FINE-GRAINED SOURCE ATTRIBUTION IN RAG-POWERED AGI RESPONSES." *International Journal of Research in Computer Applications and Information Technology(IJRCAIT)* (2024): 1104-1113. Document.

Gao, Yunfan, et al. "Retrieval-Augmented Generation for Large." *arXiv* (2024): 3-12. Document.

Lewis, Patrick, et al. "Retrieval-Augmented Generation for." *arXiv* (2021): 5-19. Document.

APPENDIX:

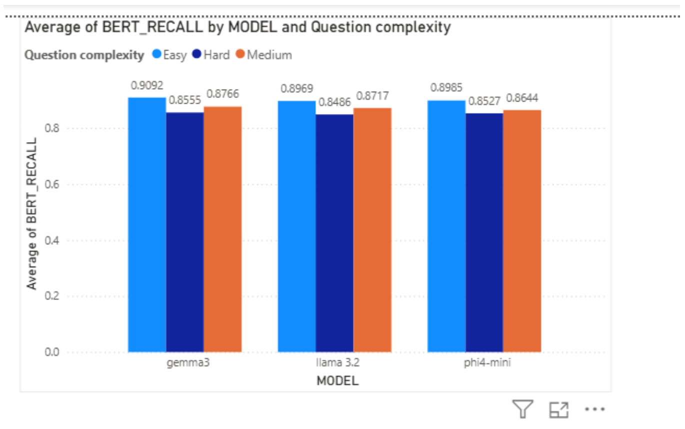


Figure 3: Avg BERT metrics for Coroner's report and blood results.

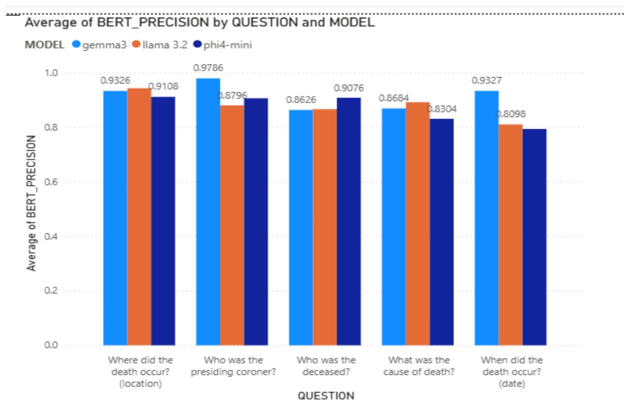


Figure 4: Avg BERT score based on question complexity.'

GROUP CONTRIBUTION TABLE:

Member name	Student number	Tasks for the project	Skills contributed to the team & project
Adam Holt	21689348	Team Leader; System Designer/Implementer; Primary Client Liaison	leadership; communication; research; system design; programming/coding
Divya Mulackal Seetharama	24523725	Done research work, did self-experiments with different components of the project and shared my ideas & knowledge gained with other team members. Designed, implemented, and coordinated the evaluation phase : (Question set & gold answers preparation, identified BERTScore as the scoring metric, organised human review, shared evaluation tasks to other team members, enforced consistency across models, and consolidated results with error analysis.)	Team collaboration & facilitation; Inclusive, collegial & participatory communication; Analytical thinking; Python; LLM Evaluation tooling; Git/GitHub workflows
Rahul Sharma	24336163	Developed a proof-of-concept PDF pre-processing pipeline with OCR (PyTesseract) for image-based reports; Managed Git/GitHub workflows, including branching, merging, pull requests, and resolving repository issues (e.g., rewriting history to remove large files); Set up and maintained a Conda environment, resolving cross-platform dependency conflicts for a stable development setup; Contributed to model evaluation by assessing Gemma3 model answers and source accuracy; Compared model performance and created visualizations for quantitative analysis.	Git & GitHub Version Control; Python Scripting; LLM Model Evaluation; Problem-Solving & Debugging; Data Visualization (Power BI)
Tahsin Rahman	24488773	Created an end-to-end evaluation pipeline to automate model evaluation. Created comparison module to distinguish between model strength and performance and find best working model; Worked on discussion and conclusion section of the report.	Effective communication, Team collaboration, Analytical Thinking, Research, Python programming

Aasish Shrestha	24583964	Prepared comprehensive technical report. Drafted the initial project proposal. Developed the core demonstration Jupyter notebook (demonstration.ipynb), porting demo.py into an interactive, well-documented guide for qanda usage.	Technical Writing & Documentation; Jupyter Notebook Development.
Zhenye Zhou	24531171	Set up and maintained development environments; ran and evaluated model tests, logged results for the team.	Reproducible environment configuration (Conda/Ollama); LLM model testing and troubleshooting.