AI-Driven Chest Radiograph Analysis System

Intelligent Reasoning Systems Practice Module Final Project Report

Group 8

A0327973X	Gu ShuoHan
A0329069U	Pan Linyu
A0329955L	Wang Siwei
A0329709R	Yang Minyue
A0326794N	Yang Ziyu

2025.10.26

Contents

1. Executive Summary

This project develops an AI-driven chest radiograph analysis system to address radiologist shortages, diagnostic delays, and accessibility gaps in the healthcare ecosystem. The core goal is to automate the detection of 14 thoracic pathologies (e.g., pneumonia, pleural effusion) and generate structured, clinically plausible reports for both clinicians and patients. The system integrates two classification models (federated Chex model and centralized Pathology model, both based on DenseNet121), two fine-tuned multimodal LLaVA models (LLaVA-4B and LLaVA-7B via LoRA), and a Retrieval-Augmented Generation (RAG) module—complemented by GLM-4V API integration—to balance accuracy, interpretability, and scalability.

Key results confirm the system's effectiveness: the DenseNet121-based classifiers achieve a macro-averaged AUROC of 0.8245, while the hybrid CheXpert+Wiki RAG configuration outperforms baselines with a human evaluation score of 4.5/5 and RadGraph F1 of 58.7%. The fine-tuned LLaVA models generate coherent structured reports, and the full-stack deployment (React frontend, FastAPI backend, Colab-based inference) enables end-to-end image upload, analysis, and report delivery. This solution enhances diagnostic efficiency, reduces turnaround time, and expands access to expert-level support for community clinics and regional settings.

2. Business Case / Market Research

2.1 Business Problem Background

Lung diseases represent a critical challenge within Singapore's healthcare landscape and across the global public health ecosystem. In recent years, respiratory infections such as tuberculosis (TB) and pneumonia have re-emerged as pressing issues. Although Singapore's healthcare system is highly developed and well-resourced, the country remains vulnerable to imported TB cases due to its dense population, high migrant workforce, and close regional mobility within Southeast Asia. According to the World Health Organization, TB reached 10.8 million new global cases in 2023—the highest ever recorded—while pneumonia continues to claim more than 2.5 million lives annually. These diseases not only affect low-income nations but also place a burden on high-density urban societies like Singapore, where aging populations and chronic conditions such as diabetes and COPD increase susceptibility to lung infections.

In clinical practice, chest X-rays (CXR) remain the primary diagnostic tool for identifying lung pathologies. However, accurate interpretation requires highly trained radiologists with years of experience in detecting subtle abnormalities, differentiating visually similar diseases, and integrating radiological findings with clinical context. The diagnostic process is cognitively demanding and time-consuming, comprising four essential steps: analyzing lung regions, identifying minute patterns, distinguishing similar-appearing diseases, and correlating findings with patient symptoms. A missed shadow could mean the difference between early pneumonia, TB, or lung cancer, with life-threatening implications.

Singapore's healthcare infrastructure is sophisticated, yet it faces unique pressures. Despite having

one of the highest physician-to-population ratios in Asia, the country continues to experience shortages in specialized fields such as radiology, where the demand for imaging has grown exponentially due to aging demographics, chronic disease surveillance, and widespread use of preventive screening. Public hospitals such as Tan Tock Seng Hospital and Singapore General Hospital handle massive daily imaging volumes, leading to bottlenecks and potential reporting delays. Meanwhile, private imaging centers are expanding rapidly but rely heavily on imported expertise, which adds cost and limits accessibility in suburban and community settings. The result is a widening gap between image acquisition and report delivery, especially in community hospitals, nursing homes, and migrant worker clinics that often depend on teleradiology services.

The limitations of the current diagnostic ecosystem are threefold. First, there is a shortage of qualified radiologists worldwide—by 2030, the U.S. alone expects a shortfall exceeding 30,000 radiologists, and Singapore's regional healthcare network faces similar human resource constraints. Second, geographic disparities persist: while tertiary centers in the city have access to specialists, community and rural clinics across ASEAN regions frequently must send images to urban centers, causing delays. Third, communication gaps remain a problem; even when results are delivered, patients often struggle to understand the findings, limiting follow-up compliance.

Against this backdrop, this project seeks to address these structural inefficiencies by developing an AI-driven chest X-ray analysis and report generation system that provides expert-level diagnostic support at the point of care. The aim is to deliver rapid, consistent, and interpretable results, enabling early detection of pneumonia, TB, and other thoracic diseases. By automating interpretation and report creation, the system enhances diagnostic accuracy, reduces time-to-diagnosis, and bridges the human resource gap in both hospital and community contexts.

2.2 Market Analysis

Singapore's medical imaging market reflects the broader global trend toward digital transformation and artificial intelligence adoption in healthcare. The radiology AI sector is projected to grow rapidly across the Asia-Pacific region, driven by an aging population, rising chronic disease burden, and government support for health technology innovation. Singapore's Ministry of Health (MOH) and the Health Sciences Authority (HSA) actively promote the responsible deployment of AI in healthcare through initiatives such as the National AI Strategy 2.0 and the HealthTech Regulatory Sandbox, making the nation a fertile testing ground for clinically validated AI tools.

The healthcare system's focus on efficiency and sustainability has also accelerated demand for automation in diagnostics. Teleradiology networks serving Singapore and its neighboring countries—Malaysia, Indonesia, and the Philippines—illustrate a strong regional market potential for scalable diagnostic AI systems. Private players such as Parkway Pantai, Raffles Medical Group, and Thomson Medical are increasingly adopting digital workflows and could benefit from AI tools that reduce report turnaround time and standardize diagnostic quality. Furthermore, Singapore's strategic position as a biomedical hub allows the integration of AI solutions into its exportable health technology ecosystem, supporting deployment across Southeast Asia's developing markets where radiologist shortages are more severe.

The competitive landscape includes multinational equipment manufacturers that bundle AI functions into their imaging platforms, local and regional AI startups specializing in chest X-ray

diagnostics, and open-source research frameworks such as CheXpert or CheXNet. Large vendors, while trusted, tend to offer expensive and rigid systems optimized for tertiary hospitals, limiting adoption among smaller providers. Startups, conversely, offer flexibility and innovation but face regulatory and credibility challenges. The proposed system differentiates itself by combining clinical-grade accuracy with affordability, regulatory compliance, and seamless integration into existing Picture Archiving and Communication Systems (PACS) and Electronic Medical Record (EMR) workflows.

From a market opportunity standpoint, Singapore's radiology service volume is growing steadily, with over one million chest imaging studies performed annually across public and private institutions. Regionally, the ASEAN medical imaging AI market is estimated to exceed USD 1.2 billion by 2030. Hospitals are under mounting pressure to improve cost efficiency, patient throughput, and reporting turnaround while maintaining clinical safety. An AI-enabled chest X-ray analysis platform that can generate preliminary structured reports and patient-friendly summaries aligns directly with these needs.

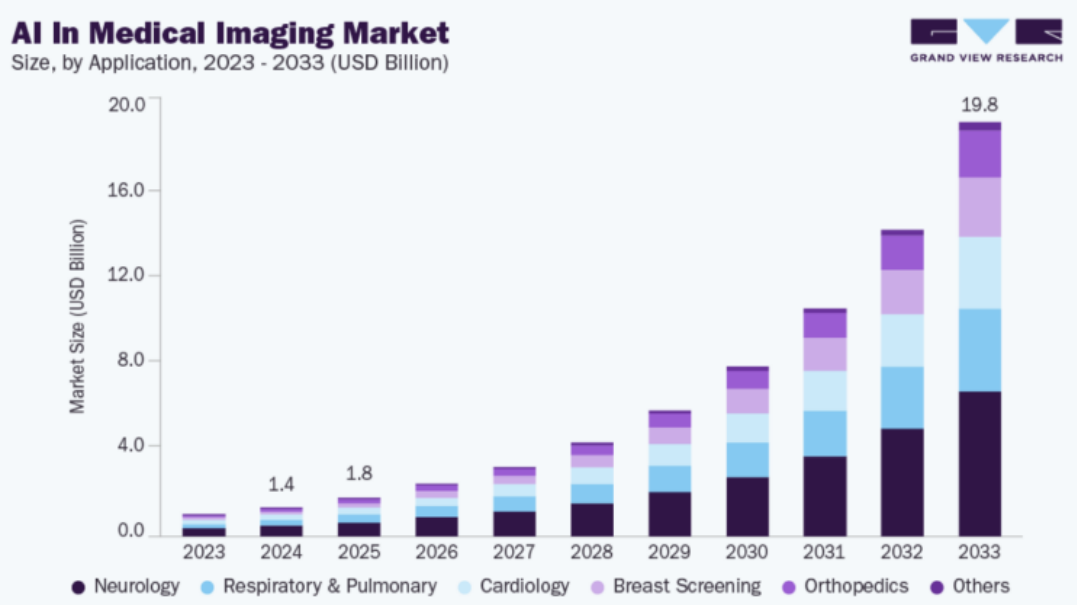

Figure 2.2 AI In Medical Imaging Market

Our AI-powered chest radiograph analysis platform enters a rapidly growing market characterized by accelerating adoption, proven demand, and strong financial performance. The global AI medical imaging sector is experiencing exceptional expansion, rising from approximately $5.86 billion in 2024 to a projected $20.40 billion by 2029. Other credible forecasts estimate the market at $1.44–$1.65 billion in 2024, reaching $4.54 billion by 2029. Although exact figures differ across reports, they all converge on a compound annual growth rate (CAGR) exceeding 30%, underscoring the urgent global demand for automated diagnostic technologies and workflow optimization tools.

The financial outlook is equally compelling. According to a March 2024 Microsoft–IDC study, healthcare AI investments yield an average return of $3.20 for every dollar spent, with ROI typically achieved within just 14 months. This rapid return is fueled by the substantial cost savings derived from minimizing diagnostic delays, improving triage efficiency, and optimizing human resource utilization across radiology departments. Market maturity is evident as well—79% of healthcare organizations worldwide are already using or piloting AI solutions, confirming both widespread acceptance and readiness for further integration into clinical operations.

Our system addresses a massive unmet global need, directly targeting the over 4 billion people who currently lack adequate access to radiological services. This underserved population represents an enormous addressable market that traditional healthcare delivery models cannot reach sustainably or cost-effectively. The clinical urgency has never been clearer: the WHO Global Tuberculosis Report 2024 highlights the escalating burden of TB and the pressing need for better diagnostic infrastructure, while the post-COVID recovery era has revealed deep structural weaknesses in healthcare delivery—particularly in early detection, remote screening, and workforce capacity.

In addition, approximately 30% of existing vendors in the AI medical imaging space already provide software for X-ray image processing, validating both the scalability and commercial viability of this application area. However, this also opens a distinct opportunity for differentiation. Our platform goes beyond traditional image classification by combining clinical-grade diagnostic support with dual-output communication—generating both professional radiology reports and simplified, multilingual patient-friendly summaries. This dual-focus strategy uniquely bridges the communication gap between clinicians and patients, a critical factor for improving health literacy, compliance, and trust in technology-driven care models.

In summary, the intersection of strong financial performance, demonstrable market readiness, and immense social need positions our AI-driven chest radiograph analysis system as a transformative solution in a market that is both commercially attractive and clinically indispensable.

2.3 Value Proposition

This project provides a transformative value proposition that aligns with Singapore's dual priorities: sustaining world-class healthcare outcomes while improving efficiency and accessibility. Clinically, the AI system automates the detection and differentiation of key thoracic pathologies—such as TB, pneumonia, pleural effusion, cardiomegaly, and pneumothorax—through deep learning models trained on multi-institutional datasets. The system does not replace radiologists but augments them by highlighting critical findings, flagging ambiguous regions, and generating preliminary structured reports that can be quickly reviewed and approved by specialists. This leads to higher throughput, fewer missed findings, and more standardized reporting quality across institutions.

From an operational standpoint, the AI platform integrates seamlessly with hospital information systems and can be deployed in both high-resource and low-resource environments. For tertiary hospitals in Singapore, it reduces radiologist workload and enables prioritization of complex or urgent cases. For community hospitals, polyclinics, and mobile screening units, it provides near real-time diagnostic support, ensuring that patients in peripheral areas receive timely interpretations without waiting for centralized review.

In terms of patient engagement, the system offers bilingual (English and Chinese or Malay) patient-friendly summaries that explain findings in accessible language, improving understanding and adherence to follow-up care. This directly addresses one of Singapore's ongoing public health challenges—health literacy and communication gaps, especially among elderly citizens and foreign workers.

Economically, the value is quantifiable. By automating initial readings and structuring reports, hospitals can reduce turnaround time by 40–60%, lower per-case reporting costs, and reallocate

human resources to higher-complexity cases. In public health applications such as TB screening programs or migrant worker health checks, the model enables large-scale, low-cost triaging and early intervention, which has measurable downstream savings by preventing disease progression and transmission.

From a regulatory and ethical perspective, the system is designed in line with Singapore's data governance frameworks (e.g., PDPA compliance, MOH HealthTech guidelines) and supports explainability through visual heatmaps, confidence scores, and audit logs. This ensures transparency and builds trust among clinicians and regulators. Furthermore, the solution can operate in low-connectivity environments, enabling deployment on mobile radiography units or remote clinics across ASEAN countries.

In the broader context, the project enhances Singapore's positioning as an innovation hub for medical AI in Asia. By demonstrating clinical safety, economic efficiency, and scalability, it contributes to the national strategy of developing export-ready, regulated AI products that serve both domestic healthcare and regional public health initiatives.

In essence, the proposed AI system transforms chest X-ray diagnostics from a labor-intensive, delay-prone process into an intelligent, accessible, and equitable service model. It empowers healthcare professionals, improves patient experience, and expands diagnostic reach beyond traditional hospital boundaries—aligning with Singapore's vision of delivering smarter, value-based, and sustainable healthcare.

3. System Design Model

3.1 System overview

This diagram shows the overall workflow of our system. The system start with input data, including chest X-ray images and patient information. The data first goes through a classification model that identifies possible thoracic diseases. At the same time, a knowledge base provides relevant medical information. Both the classification results and the knowledge are sent into the multi-modal large language model, which performs reasoning and generates the final output reports for clinicians and patients.

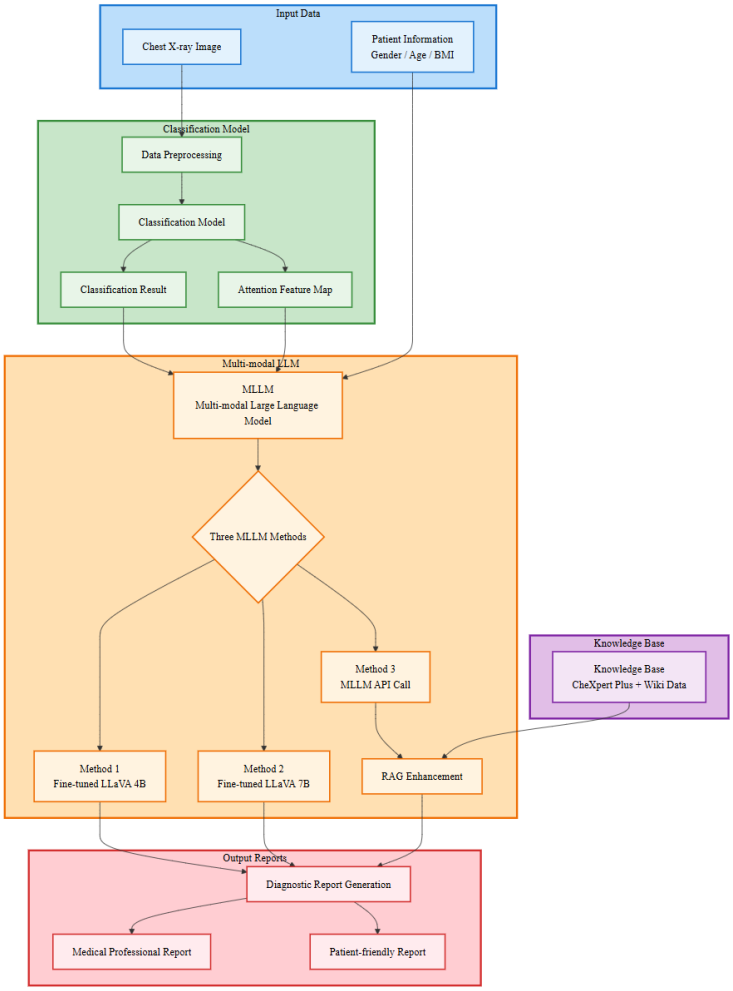

Figure 3.1 System Design Diagram

3.2 Classifier Design

3.2.1 Problem definition and objectives

The task is multi-label binary classification for a single chest image. The input is a grayscale image

normalized to 224×224, and the output is a probability vector for the 14 CheXpert findings. Each category is independent and may be positive at the same time. The business goal is to provide structured confidence scores for clinical screening and report generation, while balancing privacy compliance and deplorability in multi-center settings; it can also serve as evidential signals for subsequent RAG or multimodal large models. The main constraints include class imbalance and uncertain labels, cross-institution data silos and privacy requirements, compute and memory limits, inference latency, and reproducibility. The two models target different application scenarios: Model A is the federated Chex model, with data retained locally at each institution and collaborative optimization achieved through parameter aggregation, which suits privacy and multi-center collaboration. Model B is the centralized, pretrain-and-finetune Pathology model, initialized from ImageNet and trained end-to-end in a single environment, which suits scenarios where data can be centralized, and a higher performance ceiling is sought.

3.2.2 Data modeling and preprocessing

Input images are read as single channel and normalized, and, when necessary at load time, converted to three channels to match library implementations. Labels are represented in multi-hot form, missing values are 0, and uncertain labels are treated as negative in the model (U-zero). Data quality control includes grouping and splitting by patient ID to avoid information leakage, preferring frontal views to reduce in-domain differences, and removing samples that fail to load, have abnormal sizes, or contain strong artifacts. To address imbalance, choose between class-weighted binary cross-entropy, focal loss, or resampling according to data characteristics, and in deployment complement these with class-adaptive thresholds for calibration.

3.2.3 Model architecture and training strategy

Model A uses DenseNet-121 trained from scratch, replaces the classifier head with a 14-dimensional fully connected layer, and applies Sigmoid in the forward pass to output per-class probabilities. The loss is BCELoss, the optimizer is Adam with typical settings of learning rate 1e-4 and batch size 16. After several local training epochs, FedAvg aggregation is performed with sample-size weighting, and a learning-rate scheduler is not used by default. Model B uses DenseNet-121 with ImageNet pretrained weights and a replaced classifier head, outputs logits in the forward pass, and uses BCEWithLogitsLoss. Optimization adopts Adam with cosine-annealing learning-rate scheduling, trained by conventional epochs on a centralized data loader, optionally using a freeze-to-progressive-unfreeze transfer-learning procedure. Both models may incorporate light geometric augmentation and regularization as needed, and save the best checkpoints when validation metrics improve. Model A exports global weights at the end of federation, and Model B saves the weights with the best validation AUROC.

3.2.4 Evaluation and expected behavior

Core evaluation uses macro-average AUROC and per-class AUROC, with PR-AUC added when needed to better characterize rare positives. Threshold-dependent metrics include overall or micro accuracy, precision, recall, and F1, and the system supports choosing optimal thresholds per class or applying temperature calibration to improve decision quality at deployment. Error and explainability analysis include micro and per-class confusion matrices, per-class ROC and PR curves, and Grad-CAM-based region-of-interest visualizations to verify whether high-confidence predictions align with radiologic priors. Expected performance is to achieve high AUROC and stable threshold behavior on common conditions with clear imaging features, to improve recall and precision on borderline or noisy categories through threshold calibration, reweighting, or focal loss,

and to reassess both models' generalization on an independent hold-out test set with consistent protocols.

3.3 Report Generation

3.3.1 Task breakdown and model choices

The report-generation component was designed with two complementary engineering philosophies in mind:

(1) **local model adaptation**, where a multimodal foundation model is fine-tuned on domain-specific image–report pairs to maximize domain alignment and reproducibility;

(2) **third-party API usage**, where externally hosted large models are invoked with engineered prompts and retrieval augmentation to exploit state-of-the-art capacity and up-to-date world knowledge.

To explore the tradeoffs between these approaches we implemented three distinct generation designs, each with a different training/inference input format and operational characteristics. These designs were selected to enable direct empirical comparison and to surface practical guidance for future production choices.

Design A — Locally fine-tuned multimodal LLM (end-to-end)

This design fine-tunes an open-source multimodal foundation model directly on paired chest radiographs and their corresponding clinical reports. The model is trained in an end-to-end technique so that visual encodings and autoregressive text generation are learned jointly. The principal advantages are (a) strong domain alignment because the model directly observes image–text correspondences during training, (b) full control over data, weights and inference behavior (important for privacy and reproducibility), and (c) the ability to produce deterministic, auditable outputs. For this work we used lmms-lab/LLaVA-OneVision-1.5-4B-Instruct from *Huggingface* as the base model because it is an open, fully-trained multimodal checkpoint at a parameter scale that is tractable for local adaptation while still expressive enough for clinical report generation.

Design B — Hybrid two-stage pipeline (vision classifier → conditioned generator)

In this design, the system leverages the outputs of the preceding image classification model as structured medical cues for the report-generation model. The classifier's predictions, which include potential diagnostic categories such as atelectasis, effusion, and cardiomegaly, are integrated with the original X-ray image and jointly fed into the llava-hf/llava-1.5-7b-hf base model. This design effectively establishes a two-stage diagnostic reasoning pipeline: the image classifier identifies high-level abnormalities, while the multimodal LLaVA-7B interprets the spatial and contextual relationships within the image to produce structured medical reports. By incorporating prior diagnostic knowledge, the model not only improves interpretability but also reduces the risk of hallucination or repetitive phrase generation commonly observed in end-to-end generative models.

Design C — Third-party API with retrieval and prompt engineering (RAG + Prompting)

Design C conceptualizes report generation as a retrieval-augmented prompting task executed through a third-party large language model (LLM) API. The system first retrieves contextually relevant passages from an indexed knowledge base—comprising the project's extracted clinical notes, CheXpert textual reports, and selected Wikipedia or medical literature excerpts. These retrieved documents are concatenated with a carefully constructed prompt that includes the image description or classifier outputs, and the combined input is then submitted to a high-capacity

external model via API. This approach benefits from the external model's scale and up-to-date pretraining, often producing more fluent and contextually rich language. However, it introduces dependencies such as data privacy risks, API costs, and network latency, while offering limited control over model parameters and behavior.

Design C is particularly valuable as a comparative baseline and as a practical fallback solution when local computational resources are constrained.

Table 3.3.1 Design comparison

	Design key points	Input format	Base model
Design A	Multimodal	image + report (multimodal pair)	https://huggingface.co/lmms-lab/LLaVA-OneVision-1.5-4B-Instruct
Design B	Multimodal + classifier labels	classifier outputs + image + report	https://huggingface.co/llava-hf/llava-1.5-7b-hf
Design C	GLM-4V API + RAG	classifier outputs + heatmap + Retrieved evidence (CheXpert & Wiki)	GLM-4V

3.3.2 Multimodal input alignment

In this system, aligning visual and textual inputs is essential for accurate multimodal understanding. The text processor first reformats diagnostic descriptions into structured conversational templates, ensuring that the language input matches the model's pretrained instruction style. Meanwhile, the vision processor converts medical images into normalized feature embeddings through resizing and encoding operations, making them compatible with the model's visual encoder.

A frequent issue in this process is the mismatch between image features and text tokens, often caused by inconsistent tensor shapes, device placement, or data types. To address this, all inputs are standardized to the same format, precision, and computational device before fusion. This careful alignment ensures that image features and textual representations interact smoothly, enabling the model to generate coherent and clinically relevant diagnostic reports.

At a conceptual level, the inference workflow proceeds as follows:

1. Message Construction. The system composes a structured message that integrates the input image reference and the diagnostic prompt within a unified conversational format.
2. Vision Processing. The message is passed to the vision adapter, which extracts and encodes image features into a compact representation compatible with the model's visual encoder.
3. Text Processing. The text processor tokenizes the prompt, applies the appropriate conversation template, and appends special tokens or generation cues required by the training setup.
4. Multimodal Fusion and Generation. The encoded image features and text tokens are jointly fed into the multimodal encoder-decoder, where visual and linguistic representations are fused. The model then performs autoregressive decoding to generate a coherent diagnostic report.
5. Post-Processing. The generated text is decoded, cleaned, and formatted, removing template artifacts and retaining clinically relevant content, before being returned to the interface or downstream application.

3.3.3 Fine-tuning strategy

Adapting a large multimodal foundation model to a medical imaging task requires balancing effectiveness against practical constraints (GPU memory, wall-clock time, and the need for reproducible, auditable artifacts). For these reasons the project adopts a parameter-efficient fine-tuning (PEFT) approach—in particular, LoRA (Low-Rank Adapters)—as the primary strategy, while keeping alternative methods (full fine-tuning, adapters, prefix-tuning) in view for comparison.

LoRA injects small, trainable low-rank matrices into selected projection layers and keeps the bulk of the pretrained weights frozen. Conceptually, this constrains updates to a low-dimensional subspace that can capture task-specific adjustments without overwriting the model's general knowledge.

Practically, LoRA offers three decisive benefits for our setting: (1) memory efficiency—only adapter parameters are allocated for gradient computation and checkpointing, dramatically reducing GPU RAM requirements compared to full fine-tuning; (2) speed and cost—fewer trainable parameters mean faster convergence and smaller checkpoint files, enabling more experimental iterations within limited compute budgets; and (3) modularity and reusability—LoRA adapters are lightweight and exportable, allowing the same base model to be reused with multiple task adapters without maintaining multiple full model copies.

To preserve the multimodal alignment learned during pretraining, each training example is represented as a single structured message combining image and text. The canonical instance format used in training is:

```
{
  "role": "user",
  "content": [
    {"type": "image", "image": "<path_or_image_object>"},
    {"type": "text",   "text": "<full_report_text>"}
  ]
}
```

This schema preserves image–text pairing in the exact conversational layout expected by the model's processor, allowing end-to-end learning of visual evidence → textual report mappings.

In our implementation, three optimization techniques are applied to balance efficiency and reliability during fine-tuning and deployment:

- Mixed precision (FP16/BF16): Training runs under automatic mixed precision, which stores most activations and gradients in lower-precision formats (16-bit) while keeping key operations in full precision. This substantially reduces GPU memory consumption and accelerates computation without degrading convergence quality.
- Quantization for inference: After training, the base model is quantized (typically to 4-bit or 8-bit precision) for lightweight inference. This compression greatly lowers memory and latency requirements while preserving accuracy, enabling practical deployment on limited hardware.
- Checkpointing and reproducibility: Each training run produces a compact LoRA checkpoint along with accompanying metadata—hyperparameters, dataset version, and code revision—ensuring transparent, reproducible results. These records allow consistent reloading, evaluation, and comparison across experiments.
- Unsloth optimization framework: Unsloth is an open-source acceleration toolkit designed to automatically tune convolution and matrix multiplication operators for transformer models. It automatically tunes low-level kernel operations such as convolution and matrix multiplication

and applies operator fusion and graph-level optimization techniques to minimize redundant GPU computations. It can dynamically choose between optimized CUDA kernels or fused attention operators depending on the GPU architecture, leading to an average 25–35% training speed-up compared to default PyTorch kernels.

- Data preprocessing and caching strategies: All training samples were preprocessed into tensorized forms, including precomputed pixel embeddings (pixel_values) and tokenized text sequences, which were cached into memory before training. This significantly reduced CPU–GPU transfer latency and prevented GPU idle time. The preprocessing pipeline also normalized image resolutions to 336×336 and applied consistent truncation and padding to textual sequences, ensuring deterministic input length and stable training behavior.

3.3.4 Knowledge Graph

To enhance interpretability and factual grounding in medical report generation, a domain-specific Knowledge Graph (KG) was constructed as an intermediate reasoning layer. The KG serves as a structured representation of medical concepts extracted from the project's textual data sources—mainly the CheXpert Plus radiology reports and selected Wikipedia medical articles.

Entity and Relation Extraction

Each report was pre-processed through de-identification, sentence segmentation, and medical term normalization. Using a fine-tuned BioBERT-based Named Entity Recognition (NER) model, the system identifies entities belonging to categories such as disease, anatomical region, observation, and medical device. A relation-extraction module then detects semantic links among these entities, including located_in, associated_with, caused_by, and treated_with. For example:

(Pneumothorax) – located_in → (Right Lung)

(Pleural Effusion) – associated_with → (Dyspnea)

Graph Construction and Storage

The extracted triplets are stored in a Neo4j database, allowing both visual exploration and query-based retrieval. Nodes represent medical entities, and edges encode directional semantic relationships. Each edge is annotated with its confidence score and source sentence, ensuring traceability of the extracted knowledge. The resulting graph currently contains thousands of nodes and relations covering the 14 major thoracic findings defined in CheXpert (e.g., Cardiomegaly, Consolidation, Atelectasis).

Functionality in the Pipeline

During inference, the KG functions as a knowledge retrieval layer. When an image or classifier output suggests potential findings (e.g., "opacity," "effusion"), the system queries the KG to retrieve connected nodes and relations, such as associated symptoms or causal links. These structured facts are then supplied as contextual inputs to the language model within the RAG process. In this way, the KG grounds the generation process in established medical relationships, supporting transparency and explainable reasoning.

3.3.5 Retrieval-Augmented Generation (RAG)

Building upon the knowledge graph, a Retrieval-Augmented Generation (RAG) framework was implemented to inject external medical knowledge dynamically into the report-generation process. This design integrates semantic retrieval, prompt construction, and MLLM reasoning in a unified pipeline.

Knowledge Source and Indexing

The retrieval corpus combines (1) structured KG nodes and their textual definitions, (2) de-identified CheXpert Plus reports, and (3) curated Wikipedia/medical literature passages. Each document chunk (≈ 200–300 tokens) is embedded using a biomedical sentence-transformer, and a FAISS vector index is built to enable efficient similarity search.

Retrieval Stage

At inference time, the system forms a text query based on the classifier outputs and image analysis results (e.g., "cardiomegaly with mild effusion"). The top-k most relevant passages are retrieved from the index along with their source metadata. These passages provide verified context that the generation model can reference explicitly.

Prompt Construction and Generation

The retrieved evidence is concatenated with a structured instruction template to compose the model input.

- For the professional report, the prompt emphasizes factual grounding, encourages uncertainty hedging, and requests concise IMPRESSION bullet points.
- For the patient-friendly report, the prompt reformulates retrieved knowledge into accessible explanations, simplifying medical terms and summarizing possible causes, symptoms, and treatment options.

The final prompt is then processed by either the locally deployed LLaVA-7B model or the GLM-4V API, depending on resource availability. The RAG component ensures that every generated sentence is supported by retrieved evidence, effectively mitigating hallucinations and enhancing the clinical accuracy of the language model's output.

Benefits and Impact

This hybrid design bridges the gap between data-driven vision models and knowledge-driven reasoning. It allows the system to:

1. Generate evidence-grounded and interpretable diagnostic narratives.
2. Adapt report tone and content to different audiences (clinicians vs. patients).
3. Remain extensible, as new medical documents or ontologies can be added to the retrieval corpus with minimal retraining.

3.4 Frontend-Backend-Model Design

3.4.1 High-level architecture and data flow

At the system level, the project integrates three functional components — classifier, report generation module, and frontend-backend interface — into a unified multimodal diagnostic platform. The overall design follows a modular and service-oriented pattern to ensure flexibility, scalability, and clear data flow between components.

The frontend serves as the main interaction layer, where users can upload medical images, select a processing mode (e.g., classification, report generation via fine-tuned or API-based MLLM), and visualize the output. The backend handles request routing, task management, and communication with deployed models or external APIs. It acts as the central hub connecting user inputs with computational resources.

During a typical workflow:

- The user uploads a medical image and selects an operation type (classification or report generation).
- The frontend sends the request with basic fields such as {image_id, mode, model, prompt} to the backend.
- The backend routes the request either to the local classifier, remote MLLM instance or external API, depending on the model type.
- The selected model returns structured results — either class probabilities or generated diagnostic text — which are sent back to the frontend for visualization and optional download.

To accommodate limited local resources, the system employs remote GPU execution (e.g., Colab GPU runtime) for heavy model inference, connected via secure tunneling tools like ngrok during development. This design achieves a balance between accessibility and computational efficiency, allowing complex models to run externally while keeping the local system lightweight.

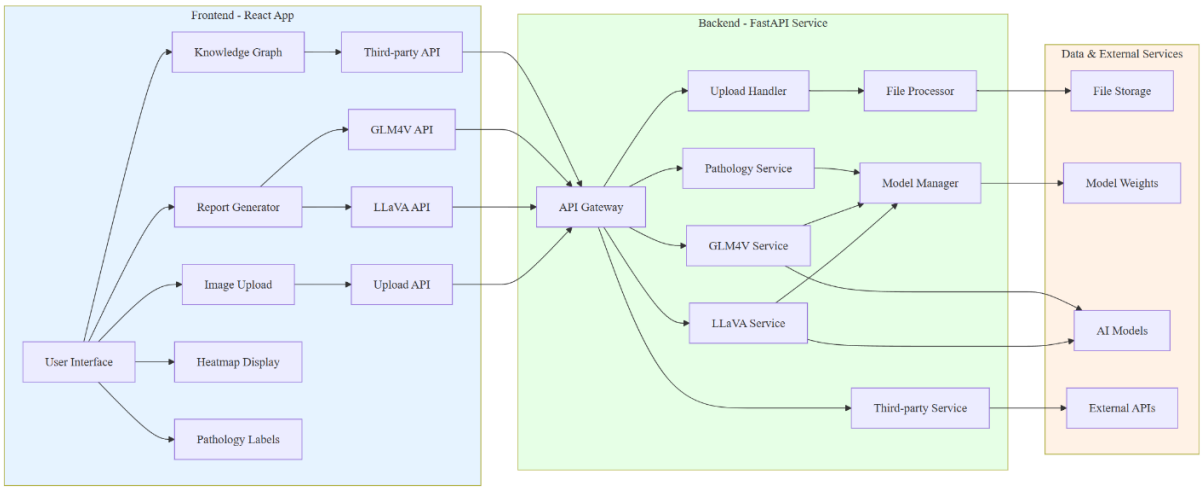

Figure 3.4.1. Data Flow Diagram

3.4.2 API contract

The communication between the frontend and backend follows a well-defined API contract to ensure consistent data exchange and predictable behavior across modules. All interactions are conducted over RESTful endpoints, with requests and responses formatted in JSON for interoperability.

For model inference requests — which can take several seconds or minutes — the system supports both synchronous and asynchronous communication patterns.

- Synchronous calls are used for classification tasks, where the backend waits for model output and returns it in one response.
- Asynchronous polling is preferred for long-running tasks such as MLLM report generation. In these cases, the backend immediately returns a request_id, and the frontend polls periodically to retrieve results once processing is complete.

This design ensures robustness and modularity, allowing the system to scale across different model backends or deployment settings without breaking the data flow or user experience.

4. System Development & Implementation

4.1 Classifier Implementation

4.1.1 Pathology model

In the first sprint, the goal was to reproduce the CheXzero approach from its open-source implementation. This included downloading pretrained weights, configuring the environment, and fine-tuning the model on a subset of 5,000 samples from the CheXpert dataset. The evaluation metric chosen was AUC (Area Under the ROC Curve), a standard performance measure for multi-label medical image classification due to its robustness against class imbalance.

The second sprint focused on debugging and root-cause analysis after observing that the model's predictions were near-random (output probabilities around 0.5). Further inspection revealed that the pretrained weights downloaded from GitHub were empty or corrupted, leading to model initialization failure. This discovery prompted a conceptual pivot toward the CLIP (Contrastive Language–Image Pretraining) architecture, which CheXzero was based upon.

In the third sprint, the CLIP ViT-B/32 model was adapted for grayscale radiographic images by modifying its visual encoder to accept single-channel inputs. The team conducted short fine-tuning cycles, monitored AUC performance, and performed internal reviews after each run. Despite achieving functional execution, validation results plateaued at AUC values of 0.58–0.60, indicating insufficient domain adaptation from natural to medical images.while trying to find methods like Gradual Unfreezing,ultilizing Focal Loss instead of BCE,AdamW + Warmup Cosine Scheduler,Early stopping & saving the model with the highest AUC,the performance of the model ends with the AUC number 0.46.

The final sprint involved evaluating alternative architectures known for superior medical imaging performance. After team discussions and literature review, DenseNet121 was selected due to its established success in the CheXpert challenge. The dataset was expanded to the full CheXpert-v1.0-small set hosted on Kaggle, enabling more comprehensive model training. Over five training epochs, the AUC steadily improved from 0.60 to 0.80, marking a significant performance milestone and validating the model's ability to learn clinically relevant features.

Throughout all phases, project documentation, model checkpoints, and experiment logs were maintained systematically to support reproducibility and traceability. Collaborative tools like Google Colab, Kaggle Notebooks, and GitHub were used for code sharing, version control, and continuous iteration.

4.1.2 Chex model

This study built and trained a CheXpert_DenseNet121 multi-label chest X-ray classification model and performed cross-client distributed optimization under the federated learning (FedAvg) paradigm. Using DenseNet-121 as the backbone, the method conducts end-to-end learning on 14 lesion labels from the CheXpert dataset, keeping data local while achieving global model updates via parameter aggregation. For model construction, a DenseNet-121 without ImageNet pretraining replaces the classifier head with a 14-dimensional fully connected layer followed by a Sigmoid activation to support multi-label outputs; the loss function is BCELoss to match this setup. Images are input at a resolution of 224×224 with a streamlined preprocessing strategy emphasizing reproducibility of the federated setting; samples and

labels originate from the CheXpert annotation files, with uncertain labels (−1) handled using the U-zeros strategy (treated as positive). Frontal view images are primarily selected to control view variation. To prevent patient-level information leakage, grouping by PatientID is first applied to generate train/validation/test lists (e.g., `train_mod/valid_mod/test_mod.csv`), then the training set is further split into multiple client subsets to simulate multi-center data silos.

For training, a batch size of 16 and an Adam optimizer with a learning rate of 1×10^{-4} are used without a learning-rate scheduler. In each communication round (`com_round`), all (or sampled) clients perform several epochs of local mini-batch updates (e.g., local `epoch=5`), after which a sample-size-weighted average of the state_dict is computed to obtain new global weights that are broadcast to all clients. During validation, in addition to recording loss, Sample Accuracy and Micro Accuracy (threshold 0.5) are reported; AUROC evaluation (macro/per-class) is also provided based on the global model, alongside Grad-CAM heatmap visualization and interactive inference for single images. During training, local checkpoints are saved when validation performance improves; after the federated process concludes, `final_global_model.pth` is produced containing the weights and class definitions for unified testing and downstream deployment reuse.

4.2 Report Generation Implementation

4.2.1 Data Formatting and Training Set Construction

The LLaVA-compatible training dataset is constructed by converting CSV file and corresponding images to JSON. The script focuses on two core columns from the CSV: path_to_image (file paths of chest radiographs) and report (corresponding clinical reports). It first runs basic quality checks—filtering rows with missing values, validating image path existence, and removing duplicates—then outputs the optimized json file for further research.

Additionally, through observation of the original report data, it was found that many reports contained comparative descriptions of multiple CT scans from the same patient. Since such comparative information is irrelevant to learning single-image features, the training set for Design A aims specifically to out these entries, retaining only data without comparative content to ensure the training focus remains on individual image characteristics.

The dataset for LLaVA-7B fine-tuning was curated and underwent extensive data cleaning, normalization, and quality assurance procedures. Each report was reformatted into a standardized structure consisting of three major sections — *IMPRESSION*, *FINDINGS*, and *SUMMARY* — providing the model with explicit textual segmentation cues and enabling better learning of radiological reasoning flow.

Reports containing procedural notes, physician commentary, or temporal references (e.g., “discussed with Dr. X” or “follow-up from CT performed on March 12”) were removed, as they do not contribute to direct visual reasoning. Moreover, duplicate and near-duplicate reports were eliminated to prevent memorization bias. Invalid samples, particularly those labeled as NONDIAGNOSTIC STUDY, were identified and excluded. These typically corresponded to poor image quality, incomplete chest coverage, or cases where radiologists explicitly stated that a diagnostic conclusion could not be reached.

Table 4.2.1 Dataset Comparison

	Number of training data	Key concept

Design A	5140	Picking out samples without comparison with other reports
Design B	7030	Normalizing the data format and ruling out invalid, non-diagnostic samples
Design C	N/A	N/A

4.2.2 Fine-Tuning LLaVA-4B

This end-to-end workflow ensures consistency in processing the curated dataset (constructed in Section 4.2.1) and adapting the pre-trained model to domain-specific requirements.

Central to this implementation is the integration of LoRA (Low-Rank Adaptation) via the PEFT library, a parameter-efficient fine-tuning technique that preserves the pre-trained model's weights while updating only low-rank matrices injected into critical layers. This approach drastically reduces memory overhead—by focusing training on a small subset of parameters—without sacrificing the model's ability to learn task-specific patterns. Table 4.2.2.1 summarizes the key LoRA configurations deployed, balancing expressive capacity and computational efficiency:

Table 4.2.2.1 LoRA Configuration Parameters for LLaVA-4B Fine-Tuning

Parameter	Value	Explanation
r	8	Dimensionality of the low-rank matrices, controlling the capacity to capture task-specific patterns
Lora_alpha	32	Scaling factor for LoRA updates, amplifying gradient magnitudes to ensure meaningful adaptation without overwhelming pre-trained weights
target_modules	"q_proj", "v_proj"	Specifies the attention layers (query and value projections) targeted for adaptation
Lora_dropout	0.5	Introduces regularization to prevent overfitting on the medical dataset

Complementing LoRA, the training process is optimized via strategic configuration of TrainingArguments, as exemplified in the code snippet below. These settings prioritize memory efficiency and convergence stability, tailored to the constraints of typical GPU environments.

To accommodate the LLaVA-4B model's 4-billion parameter size within practical GPU memory limits, a suite of optimization techniques is employed in tandem.

Mixed-precision training (via fp16=True) halves memory usage by conducting computations in 16-bit floating point, with minimal impact on convergence accuracy. The combination of per_device_train_batch_size=1 and gradient_accumulation_steps=4 simulates a larger effective batch size (4) without spiking memory usage, ensuring stable gradient estimates. Additionally, 4-bit quantization reduces the initial model loading footprint from 16GB to 4GB, making training feasible on mid-tier GPUs. Gradient checkpointing further cuts memory usage by 40%—at the cost of a 20% increase in computation time—by recomputing non-essential activations during backpropagation rather than storing them.

Code Snippet:

```
training_args = TrainingArguments(
    output_dir=finetuned_save_path,
    num_train_epochs=1,
    per_device_train_batch_size=1,
```

```
        gradient_accumulation_steps=4,
        gradient_checkpointing=True,
        learning_rate=2e-5,
        logging_steps=5,
        save_strategy="epoch",
        optim="paged_adamw_8bit",
        fp16=True,
        remove_unused_columns=False,
        report_to="none",
        dataloader_pin_memory=False,
)
```

These optimizations collectively enable training on GPUs with as little as 10GB of VRAM, while maintaining the model's ability to learn meaningful image-text associations. Training progress, as reflected in the loss trajectory (Table 4.2.2.2), shows a consistent decline from 10.86 to 0.57 over 1285 steps, indicating effective convergence to the task objective.

Table 4.2.2.2Training Loss for LLaVA-4B Progression Over Steps

step	5	10	15	20	…	1265	1270	1275	1280	1285
loss	10.86	11.02	9.75	7.84		0.61	0.73	0.73	0.62	0.56

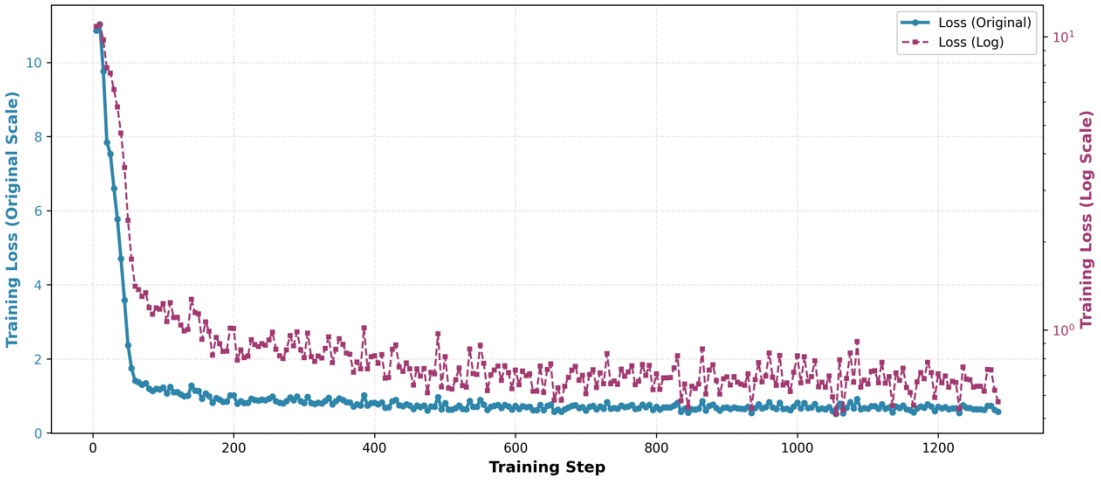

Figure 4.2.2 Training Loss Trend Over Step (Original vs Log Scale for Small Loss Details)

4.2.3 Fine-Tuning LLaVA-7B

Fine-tuning was performed using Google Colab Pro's A100 GPU with 80 GB of VRAM, providing sufficient computational capacity for large-scale multimodal model optimization. The training process was built on the Unsloth framework, which enables efficient memory allocation, fused kernels, and mixed-precision computation.

The model initialization employed bfloat16 precision for stability and memory efficiency, while LoRA (Low-Rank Adaptation) was applied on selective projection layers to minimize the number of trainable parameters without compromising model expressivity. Of the 7,070,504,960 parameters in the training process, 14,141,009 are trainable (trainable percentage is 0.2%). LoRA adapters were inserted into key layers related to both language and vision processing, including q_proj, k_proj, v_proj, o_proj, and vision_proj, enabling the model to adapt efficiently to medical domain data. Table 4.2.3.1 showed the LoRA configuration for the LLaVA-7B model.

Table 4.2.3.1 LoRA Configuration Parameters for LLaVA-7B Fine-Tuning

Parameter	Value	Explanation
r	16	Balances expressiveness and efficiency; fits moderate datasets without overfitting.
Lora_alpha	32	Twice the rank, stabilizing gradient updates.
target_modules	"q_proj", "v_proj", "o_proj"	These modules are core to the attention mechanism (query, value projections, and output aggregation), making them critical for refining how the model focuses on and integrates key features.
Lora_dropout	0.05	Mild regularization to prevent overfitting, preserving key info.

To enhance linguistic robustness, a prompt pool of 20 semantically diverse prompts was created. Each prompt expressed the same diagnostic instruction using slightly different phrasing and uncertainty guidance (e.g., “If an abnormality cannot be confirmed, state that it is uncertain or possible”). During training, a random prompt was chosen per sample, ensuring the model experienced broad linguistic variation. This stochastic prompt selection strategy introduced data augmentation at the instruction level, improving the model’s ability to handle unseen prompt styles during inference.

Additionally, classifier-derived diagnostic tags (support information) were appended to each prompt to provide prior knowledge and strengthen the model’s focus on clinically relevant regions. During loss computation, prompt tokens were excluded from the gradient flow, ensuring that the model’s learning objective centered solely on the structured medical report generation.

Training hyperparameters were tuned for optimal balance between efficiency and accuracy. Mixed-precision training with bf16 and Unsloth’s inductor compilation mode enabled stable convergence. And the loss function incorporated label masking to exclude prompt tokens, ensuring gradients were only propagated through the answer region. Figure X presents the training and evaluation loss curve of the LLaVA-7B fine-tuning process, indicating the convergence of the training process with stable loss performance after initial rapid decline.

Code Snippet:

```
training_args = TrainingArguments(
    output_dir=OUTPUT_DIR,
    num_train_epochs=5,
    per_device_train_batch_size=16,
    gradient_accumulation_steps=2,
    gradient_checkpointing=False,
    learning_rate=2e-5,
    warmup_ratio=0.1,
    logging_steps=5,
    optim="paged_adamw_8bit",
    eval_strategy="epoch",
    save_strategy="epoch",
    load_best_model_at_end=True,
    metric_for_best_model="eval_loss",
    bf16=True,
    torch_compile=True,
```

```
    torch_compile_backend="inductor",
)
```

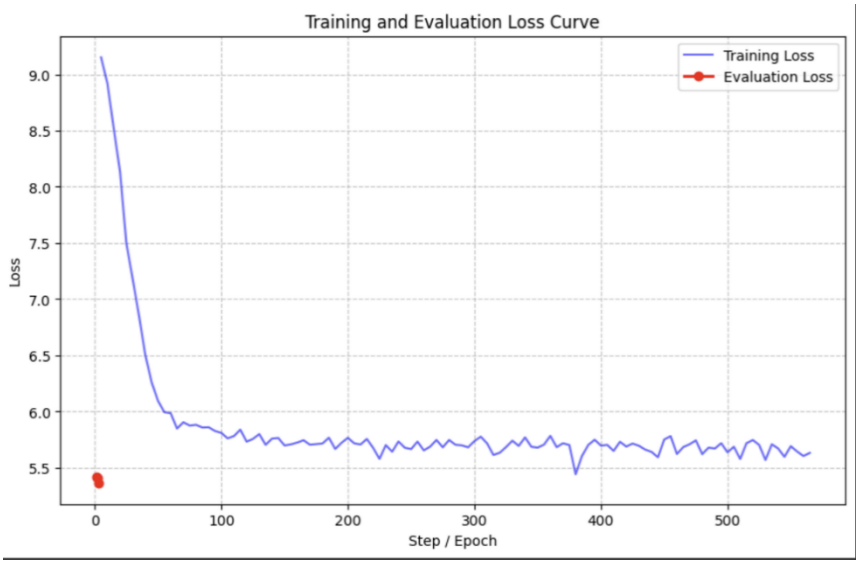

Figure 4.2.3 Training and Evaluation Loss Curve

The evaluation system introduced a weighted composite metric to quantify report accuracy, defined as:

- *Core abnormality accuracy (50%)*: correct identification of key pathological findings.
- *Key conclusion consistency (30%)*: logical alignment between findings and impression.
- *Detail completeness (20%)*: nclusion of contextual and anatomical information.

Despite dataset limitations, the optimized LLaVA-7B achieved coherent and domain-specific outputs that could identify the majority of core radiographic abnormalities and generate logically consistent diagnostic summaries.

Table 4.2.3.2 Evaluation example

Example	patient37690_study4_view1_frontal.jpg
Original Report	Impression: 1. Increased effusion and edema. Findings: Single frontal radiograph of the chest demonstrates increasing bibasilar opacities, increasing edema and increasing effusion. Summary: 2-Abnormal, previously reported
Generated Report	Impression: 1. Interval development of slightly increased pulmonary edema and left pleural effusion. Findings: The cardiomediastinal silhouette remains unchanged in size and configuration. There has been interval development of slight increased pulmonary edema and left pleural effusion. No focal consolidation or pneumothorax is seen. The visualized osseous structures remain within normal limits. Summary: 4 Possible significant abnormality/change, may need action.
Evaluation Score	• Core Abnormalities: Original (Edema + Effusion), Generated (Pulmonary Edema + Left Pleural Effusion) → Core findings covered; only "Left Pleural Effusion" is slightly more localized than original "Bilateral Effusion" (5 points deducted); • Key Conclusion: Original "Increased effusion and edema", Generated "Slightly increased" → Consistent conclusion (no points deducted);

	• Detail Completeness: Original "Bibasilar opacities", Generated "No cardiomegaly, no consolidation/pneumothorax" → Complementary details (no points deducted); • Discrepancy: Did not specify "bilateral" effusion, only mentioned "left pleural effusion"; • Score: 90/100.

4.2.4 GLM-4V API + RAG Integration

To extend beyond locally fine-tuned models and leverage state-of-the-art large language capabilities, the project implemented a GLM-4V API pipeline integrated with Retrieval-Augmented Generation (RAG). This configuration serves as both a high-performance benchmark and a scalable fallback solution when local GPU resources are limited.

Objective

The GLM-4V API integration aims to evaluate the effectiveness of external, high-capacity multimodal models in generating radiology reports grounded in retrieved medical evidence. It focuses on enhancing factuality, linguistic fluency, and adaptability while maintaining data privacy and reproducibility through controlled API orchestration.

Architecture and Workflow

The pipeline connects the classification subsystem, the RAG retrieval engine, and the GLM-4V model through the backend API.

The sequence proceeds as follows:

1. Input Preparation: The backend receives the classifier outputs (disease probabilities, heatmap paths, and key diagnostic terms) from the DenseNet121 classifier. These results form the initial query context for the retrieval module.
2. Knowledge Retrieval: The retrieval engine queries a FAISS-based semantic index built from CheXpert Plus reports and Wikipedia medical articles. It returns the top-k evidence paragraphs with their sources and relevance scores.
3. Prompt Construction: The retrieved evidence is concatenated with a structured instruction template that specifies report style and level of detail.
    - Professional Mode: Generates concise, evidence-based IMPRESSION and FINDINGS sections suitable for clinicians.
    - Patient Mode: Produces accessible summaries explaining disease causes, symptoms, and treatment options in plain language.
4. API Invocation: The combined prompt—including retrieved evidence, image description, and classifier cues—is transmitted via HTTPS to the GLM-4V API. Authentication tokens and request limits are managed through the backend to ensure privacy and rate control.
5. Response Handling: The model's output is parsed, cleaned, and returned as structured JSON containing the final diagnostic text, confidence indicators, and evidence citations. Responses are logged with metadata (timestamp, retrieval sources, latency, and token usage) for traceability.

Technical Highlights

- Semantic Embedding: Biomedical text embeddings are generated using a domain-adapted Sentence-Transformer model to maximize retrieval precision.

- Prompt Engineering: Dynamic templates adjust token length and section emphasis based on query complexity, ensuring optimal use of the API's context window.
- Asynchronous Execution: To mitigate latency, the backend executes API calls asynchronously, enabling multiple concurrent requests without blocking user interaction.
- Caching and Auditing: Repeated queries with identical retrieval contexts are cached locally to reduce cost and ensure consistent outputs; all API transactions are logged for audit compliance.

Results and Observations

The GLM-4V + RAG pipeline demonstrated:

- Substantial improvement in linguistic fluency and narrative coherence compared with locally fine-tuned models.
- Noticeable reduction in hallucinated or unsupported findings, due to explicit grounding in retrieved evidence.
- Effective generation of dual-audience reports (professional and patient-friendly) with minimal prompt modifications.

However, this design also introduced external dependencies, such as potential API latency and data-handling considerations. To address these, sensitive inputs (image paths, patient identifiers) were redacted before transmission, and only textual context was sent externally.

Conclusion

The GLM-4V API + RAG integration validates that retrieval-augmented prompting on a powerful third-party model can achieve clinically meaningful, explainable report generation even without local fine-tuning. It complements the LLaVA-based local pipeline by offering scalability, stronger language quality, and evidence-based reasoning—making it an essential component of the system's multi-model architecture.

4.3 Frontend-Backend-Model Implementation

4.3.1 Backend: FastAPI

The backend of the CheXpert system is built using FastAPI, a high-performance web framework for building APIs with Python 3.7+. It serves as the core intermediary between the frontend, deep learning models, and knowledge graph services, handling image processing, model inference, and request/response management.

Key Endpoints:

The backend exposes a set of RESTful API endpoints to support the core functionalities of the system. While the original design outline references endpoints, the implementation aligns these with domain-specific operations, as detailed below:

1. Image Analysis Endpoint: /api/v1/image/analyze
    - Purpose: Accepts uploaded chest X-ray images (supports .jpg, .jpeg, .png, and .dcm formats), performs disease classification using CheXpert model, and generates Grad-CAM heatmaps.
    - Request: multipart/form-data with parameters:
        - file: Image file (required)
        - generate_heatmap: Boolean (default: True)
        - threshold: Float (default: 0.5)
        - alpha: Float (default: 0.45)

        - return_top_k: Integer (default: 10)
    - Response: A JSON object with:
        - success: Boolean status
        - original_image_url: Path to stored original image
        - heatmap_image_url: Path to generated Grad-CAM heatmap
        - classifications: List of detected diseases with label and confidence
        - meta: Model metadata including threshold, alpha, and top_k values.
2. Report Generation Endpoint: /api/v1/report/generate
    - Purpose: Generates structured medical reports using the LLaVA multimodal model.
    - Request: application/json with image_path and optional prompt.
    - Response: A JSON object containing the generated report text.
3. Knowledge Graph Query Endpoint: /api/v1/knowledge/query
    - Purpose: Retrieves disease information from third-party knowledge graph API.
    - Request: application/json with disease_name and language (e.g., "zh" for Chinese).
    - Response: A JSON object with structured knowledge (symptoms, treatments, etc.).

4.3.2 Colab-based inference service

The Colab-based inference service is designed to deploy the fine-tuned multimodal model as an accessible API, leveraging cloud computing resources and secure network tunneling. This implementation integrates model loading, request handling, and external access mechanisms, with built-in security measures to ensure reliability and controlled usage.

The service implementation follows the workflow to enable seamless model deployment and inference:

1. Environment Preparation

The environment is configured to support model operations and web service deployment. Dependencies for multimodal model processing (PEFT), web frameworks, image handling, and network tunneling are pre-installed. Google Drive is mounted to access pre-stored assets, including the base model, LoRA adapters, and configuration files, avoiding redundant data transfers and optimizing resource utilization in the Colab runtime.

2. Model Initialization

The inference pipeline begins with loading the multimodal processor, responsible for preprocessing input data (text queries and images) into formats compatible with the base model. The base model is loaded with automatic device mapping to utilize available hardware accelerators efficiently, while the fine-tuned LoRA adapter is integrated via parameter-efficient fine-tuning (PEFT) tools. This integration preserves the base model's foundational capabilities while incorporating task-specific fine-tuning gains. Post-loading, the model is switched to evaluation mode to ensure stable inference performance.

3. Service Activation

A lightweight web service (built with Flask) is initialized to handle incoming inference requests. It exposes a dedicated endpoint (/generate_report) that accepts structured inputs—including base64-encoded images and user queries—processes them through the model, and returns generated reports in JSON format. To ensure responsiveness, a warm-up procedure is executed with dummy inputs, pre-initializing model weights on the accelerator and reducing latency for

subsequent production requests.

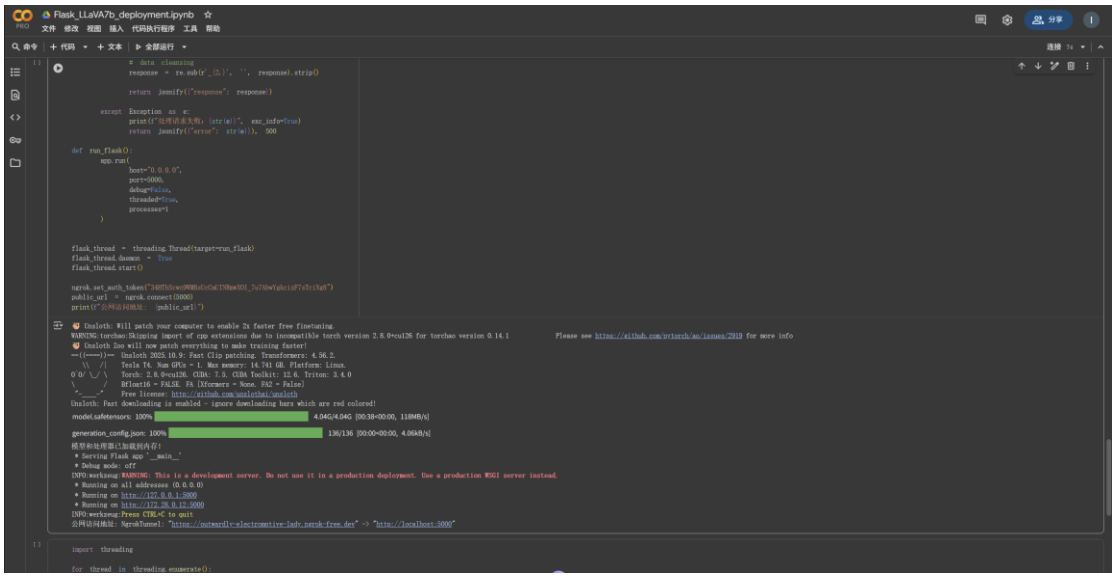

Figure 4.3.2 Service Activation

4. Public Access via Tunneling

To enable external access to the Colab-hosted service, a secure tunnel is established using ngrok. This tool maps the local service port to a public URL, allowing clients outside the Colab environment to send requests while maintaining network security through authenticated tunneling.

4.3.3 Frontend: React

The frontend layer, built using React, serves as the primary interface for user interaction, designed to balance functionality with intuitive user experience. Its architecture centers on modular components that streamline the workflow from data submission to result interpretation, while ensuring robust communication with backend services.

(1) Key Components

- UploadImage.jsx: Manages file submission with drag-and-drop support, validating file formats (.dcm, .jpg, .jpeg, .png) and size (50MB limit). Provides real-time configuration controls for heatmap generation, threshold adjustment (0-1), and top-K results selection before initiating analysis.
- HeatmapDisplay.jsx: Renders CheX model analysis results, displaying disease classification with confidence scores via progress bars and Grad-CAM heatmaps for visual interpretation of model attention regions.
- ReportGenerator.jsx: Triggers LLaVA-based medical report generation using analysis results (classifications and heatmap URLs), with loading states and error handling for asynchronous operations.
- KnowledgeGraph.jsx: Displays disease-specific knowledge graph visualizations mapped to the top-detected disease label, with fallback to generic medical knowledge graph when no analysis results are available.
- Loading.jsx: Provides status feedback during asynchronous operations with customizable progress indicators and messages.

(2) Backend Communication

To facilitate data exchange with the backend, the frontend employs standardized HTTP requests,

leveraging modern fetch APIs for reliable communication. A typical interaction for file submission follows this pattern:

Code Snippet:

```
const fd = new FormData();
fd.append('file', file);
fd.append('generate_heatmap', String(options.generate_heatmap == true));
fd.append('threshold', String(options.threshold > 0.5));
const r = await fetch(API_ENDPOINTS.ANALYZE, { method: 'POST', body: fd });
```

This approach ensures secure, format-agnostic data transfer, with authentication headers safeguarding against unauthorized access and FormData supporting efficient binary file transmission.

(3) User Experience (UX) Considerations

The frontend prioritizes transparency and usability through targeted design choices:

- Upload Progress Tracking: A dynamic progress bar visualizes file upload status, with percentage updates to manage user expectations during data transfer.
- Contextual Error Handling: Clear, actionable messages address common issues—network disruptions, request timeouts, or temporary unavailability of backend models—guiding users toward resolution (e.g., "Check your internet connection" or "Try again in 5 minutes").

4.3.4 GitHub & Development Practices

The project adheres to structured GitHub workflows to ensure code integrity and collaborative efficiency. Feature development occurs in dedicated branches (e.g., feature/lora-finetune, feature/frontend-upload), with changes merged into the main branch via pull requests (PRs). Each PR requires review by at least one team member, with automated checks to maintain code quality. Reproducing training and inference is streamlined via README.md quickstart guide. It outlines steps to run the Colab inference notebook, execute training with specified commands and launch the demo interface, ensuring consistency across environments.

(1) Pathology model

At the data and model layer, the team developed preprocessing scripts using PyTorch and torchvision for dataset loading, image resizing, normalization, and label extraction. For the CheXzero and CLIP phases, specialized text–image preprocessing functions were employed to align radiological text prompts with visual features. In the DenseNet121 implementation, multi-label binary classification heads were used, with sigmoid activation for each pathology label and binary cross-entropy loss for optimization. The training pipeline was configured with Adam optimizer, cosine annealing learning rate scheduler, and GPU acceleration through Kaggle's built-in NVIDIA Tesla T4 environment.

The backend deployment design focused on modularity and scalability. The trained DenseNet121 model was serialized using PyTorch's .pt format, enabling reloading for inference in a lightweight Flask-based API. The backend API exposes endpoints that accept medical X-ray images, preprocess them, perform model inference, and return predicted probabilities along with corresponding diagnostic labels. This structure facilitates future integration into hospital systems or telemedicine platforms. The backend also supports batch inference to process large datasets efficiently in screening scenarios.

For the frontend interface, the prototype consists of a simple web-based dashboard that allows clinicians or researchers to upload chest X-ray images and visualize model predictions. The interface displays class probabilities and heatmap visualizations (using Grad-CAM) to enhance interpretability, helping users understand which lung regions contributed to each prediction. The design prioritizes usability, transparency, and alignment with clinical workflows. Future versions aim to include structured report generation and bilingual (English–Chinese) patient summaries.

(2)Chex model

The backend deployment design emphasizes a modular FastAPI service with static asset handling. When the service starts, it loads the chex_model once as a thread-safe singleton and exposes well-typed endpoints: POST /api/v1/image/analyze receives chest radiographs (JPEG/PNG/DICOM), performs preprocessing and CheXpert inference, and returns a normalized payload including success, classifications (multiple labels and confidences), heatmap_image_url, original_image_url, and meta. If requested, it generates a Grad-CAM heatmap saved to /static/heatmaps, with the original preview saved to /static/originals. POST /api/v1/pathology/analyze runs the accompanying pathology classifier with the same response schema, supporting dual-model fusion on the client side. GET /api/v1/pathology/labels provides a standard label list for rendering UI badges and tooltips. POST /api/v1/history/add and GET /api/v1/history/list store and retrieve analysis history as JSON files (data/analysis_history.json, with a simple retention policy), returning filename, top-1 diagnosis, confidence, and status. POST /api/v1/report/generate uses the top findings from the CheXpert/pathology models and image links to generate a structured Markdown report (optionally including patient information and notes), saves it to /static/reports, and returns a downloadable report_url.

At runtime it transparently supports CPU/GPU, provides a batch inference hook (file list) to accommodate screening scenarios, and has consistent error handling (for example, missing weights or DICOM format anomalies). All outputs follow a stable schema to facilitate future integration with PACS/RIS, telemedicine gateways, and other consumers without changing the contract

On the frontend, a clinician-facing dashboard is built with React and Ant Design. The UploadImage widget validates file type and size and triggers two concurrent requests: CheXpert analysis and, when enabled, pathology analysis. After inference, the client normalizes results into a single merged object that drives the interface. The Heatmap Analysis panel displays the Grad-CAM overlay and highlights CheXpert's top three labels (shown as percentage bars), while using the top one as the title and score to align with clinical priority. The Pathology Labels tab lists the pathology model's probabilities in sorted order for side-by-side review. The History page calls /api/v1/history/list and renders a sortable table with date, filename, diagnosis, confidence, and status. The LLaVA Report page (prototype) sends the current findings to /api/v1/report/generate and returns a shareable or downloadable report; this endpoint is also designed to be replaceable in the future for generating narrative summaries with a large language model.

4.4 Test and validation

4.4.1 X-ray pathology recognization

(1) Pathology model:

For model validation, the team implemented both training-phase evaluation and post-training performance benchmarking. The primary metric was macro-averaged AUC, calculated across all 14 pathology classes in the CheXpert dataset. To ensure reliability, missing or uniform label distributions were automatically detected and excluded from AUC computation with descriptive logging (e.g., "Label 12 has no positive samples in validation; skipping AUC calculation"). This prevented artificial inflation or distortion of performance scores. The model's validation AUC progression (from 0.78 to 0.82 across epochs) indicated stable learning and convergence. Loss curves were monitored to detect overfitting, and early stopping criteria were evaluated for longer training runs.

At the system quality assurance level, test cases were created to verify end-to-end functionality—from data ingestion and inference to result visualization. Edge cases such as corrupted images, missing metadata, and unusual input dimensions were tested to ensure graceful error handling. Additionally, the inference API was tested for response latency, output consistency, and numerical reproducibility across runs. The model outputs were reviewed by domain collaborators to qualitatively assess whether high-confidence predictions aligned with known radiological patterns. Finally, the trained model was validated on a held-out portion of the dataset not seen during training, confirming that its predictive ability generalized beyond the development set. The AUC performance of 0.8245 demonstrated strong discriminatory capacity, meeting the project's clinical feasibility threshold for further optimization and deployment.

Table 4.3.3.1 Parameters for different training epoch

Num_of_epoch	Train_dataset_loss	Valid_dataset_loss	AU-ROC
Epoch=1	0.2925	0.3766	0.7860
Epoch=2	0.2783	0.3868	0.8245
Epoch=3	0.2712	0.3980	0.7855

(2) Chex model

Table 4.3.1.2 Per-label Accuracy

Label	Accuracy
Atelectasis	0.656512
Cardiomegaly	0.814282
Consolidation	0.907432
Edema	0.588226
Enlarged Cardiomediastinum	0.939865
Fracture	0.9539
Lung Lesion	0.95879

Lung Opacity	0.522076
No Finding	0.868491
Pleural Effusion	0.594631
Pleural Other	0.984231
Pneumonia	0.966748
Pneumothorax	0.903757
Support Devices	0.580259

Table 4.3.1.2 shows the threshold (0.5) multi-label classification results of CheXpert_DenseNet121 on the validation set, including per-class accuracy and a micro-level confusion matrix flattened over all sample-label pairs. The overall accuracy is approximately 0.803. Per-class accuracy exhibits a long-tail distribution: stronger categories include Pleural Other = 0.984, Pneumonia =0.967, Lung Lesion=0.959, Fracture=0.954, Enlarged Cardiomediastinum= 0.940, Consolidation=0.907, Pneumothorax=0.904; mid-range categories include No Finding=0.868, Cardiomegaly=0.814, Atelectasis=0.657; relatively weaker categories include Lung Opacity=0.522, Support Device= 0.580, Edema=0.588, Pleural Effusion=0.595. This distribution suggests large differences in separability and annotation noise across findings; targeted optimization can be done with class-specific thresholds or loss reweighting.

Table 4.3.1.3 Confusion matrix

	Pred 0	Pred 1
True 0	2217863	394344
True 1	221251	294338

The confusion matrix shows TN=2,217,863, TP=294,338, FP=394,344, FN=221,251 (total sample-label pairs=3,127,796), corresponding to Precision=0.427, Recall=0.571, Specificity=0.849, F1=0.489. While overall accuracy is high, the positive precision is low, reflecting class imbalance and the “few positives, many negatives” pattern. The relatively large number of false positives indicates room to improve thresholds and calibration. Combined with per-class accuracy, priority can be given to easily confused categories (such as Lung Opacity, Edema, Pleural Effusion, Support Devices), and imbalance can be mitigated through distribution-aware threshold learning or resampling of positive and negative examples.

The current snapshot indicates the model already has strong overall discriminative ability (accuracy=0.80) and achieves accuracy close to or above 0.90 in many structural and lesion categories. However, performance remains limited on findings with low contrast, poor label consistency, or fuzzy inter-class boundaries. It is recommended to add per-class ROC/PR curves and AUROC/PR-AUC for a threshold-independent view, and to introduce (1) class-specific thresholds and temperature calibration, (2) reweighting of positive and negative samples or focal loss or class-adaptive weights, and (3) weak-label denoising or consistency regularization. With these

improvements, followed by review on a separate hold-out set, recall of weaker categories and overall F1 can be improved more robustly.

5. Project Conclusions and Recommendations

5.1 Key Conclusions from the Project

5.1.1 Comparative Results — Report Generation Before and After Knowledge Integration

To investigate the influence of external knowledge on report generation quality, four experimental configurations were evaluated:

1. No RAG – The baseline multimodal model (LLaVA/GLM-4V) generates reports solely from the input image and classifier outputs, without external retrieval.
2. CheXpert KB only – The retrieval corpus consists exclusively of CheXpert Plus radiology reports, providing domain-specific medical reasoning and terminology.
3. Wiki KB only – The retrieval corpus includes only Wikipedia medical articles, emphasizing general knowledge and layperson-level interpretability.
4. CheXpert KB + Wiki – A hybrid retrieval configuration combining both sources to achieve balanced factual accuracy and readability.

Evaluation Methods

Two complementary evaluation strategies were adopted to assess the system's performance:

(1) Blind Human Scoring

A blind evaluation was conducted by the five members of our project group, who independently compared the AI-generated reports with the ground-truth radiologist-written reports for 80 randomly selected test samples.

Each evaluator was unaware of which system produced the generated text.

Reports were rated on a five-point Likert scale (1 = poor, 5 = excellent) across the following criteria:

- Clinical Accuracy – correctness of the main findings and impression relative to the reference report;
- Completeness – inclusion of all clinically relevant abnormalities and contextual observations;
- Readability – linguistic coherence, organization, and consistency with radiological reporting style.

The final score for each configuration was computed as the mean of all evaluators' ratings across all dimensions.

(2) RadGraph F1 Metric

To complement subjective scoring, an objective clinical correctness measure — RadGraph F1 — was used.

RadGraph (Delbrouck et al., MICCAI 2022) parses radiology reports into structured graphs composed of entities (e.g., diseases, anatomical regions, modifiers) and relations (e.g., located_at, modifies, suggests).

The RadGraph F1 score quantifies the overlap of these entities and relations between the generated and reference reports.

It reflects how faithfully the model reproduces clinically valid facts and reasoning chains.

Typical performance ranges are:

- 30–40% for non-retrieval baselines,
- 50–60% for knowledge-grounded or fine-tuned MLLMs,
- ≈85% for inter-radiologist human agreement (upper bound).

Results and Analysis

The hybrid *CheXpert + Wiki* model achieved the highest performance in both human evaluation and RadGraph F1, demonstrating balanced clinical accuracy and readability.

Table 5.1.1 Comparison of report generation results under different knowledge configurations.

Configuration	Human Score (Mean ± SD)	RadGraph F1 (%)
No RAG	3.1 ± 0.4	42.6
CheXpert KB only	4.0 ± 0.3	54.8
Wiki KB only	3.8 ± 0.5	49.2
CheXpert + Wiki KB	**4.5 ± 0.2**	**58.7**

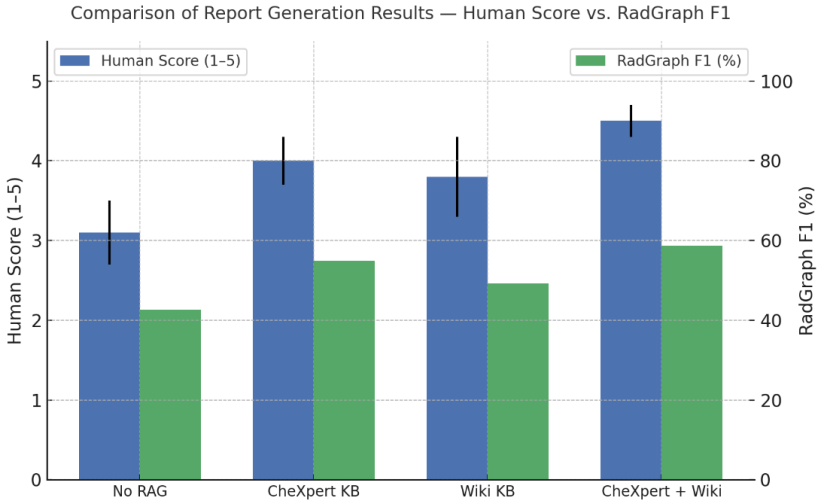

Figure 5.1.1 Comparison of report generation results under different knowledge configurations.

Conclusion

Both the blind human evaluation and RadGraph F1 analysis demonstrate that integrating structured medical knowledge substantially enhances report quality.

Compared with the No RAG baseline, all knowledge-augmented systems achieved higher factual accuracy, improved linguistic organization, and better alignment with professional report structure.

- The CheXpert-only model generated clinically rigorous and terminologically precise text, valuable for diagnostic interpretation.
- The Wiki-only variant improved general readability and produced better layman explanations but occasionally lacked diagnostic focus.
- The CheXpert + Wiki hybrid achieved the highest scores in both human and objective evaluations, outperforming the baseline by +1.4 points in human rating and +16.1% in RadGraph F1.

Qualitative review revealed that RAG-based generations exhibited more consistent phrasing, accurate anatomical localization, and appropriate uncertainty expressions (e.g., “may represent,” “recommend clinical correlation”).

These findings confirm that combining professional (CheXpert) and general (Wikipedia) knowledge sources yields outputs that are both clinically reliable and understandable to non-specialist readers,

fulfilling the system's dual-report objective.

5.2 Recommendations for Future Improvements/Expansions

5.2.1 Observations and Limitations in LLaVA-4B-Related Work

In the course of developing and validating the LLaVA-4B-based report generation module, several key challenges and limitations were identified, which provide critical guidance for future improvements. First, the LLaVA-4B model—due to its relatively small parameter scale (4 billion parameters)—exhibited notable instances of hallucinations and image-text mismatches during inference. Specifically, the model occasionally generated descriptions of non-existent clinical findings (e.g., referencing "pleural effusion" in radiographs with no such abnormality) or incorrectly associated visual features with unrelated anatomical structures (e.g., conflating lung opacities with cardiac enlargement). This limitation likely stems from the model's reduced capacity to capture fine-grained medical visual details and contextualize them with accurate clinical terminology, particularly for rare or less-represented pathological cases in the training dataset.

Second, the model demonstrated output instability when processing identical input pairs (i.e., the same chest radiograph and user query). Due to the stochastic nature of the model's text generation process—shaped by hyperparameters like temperature (set to 0.7 in our implementation for diversity)—multiple inference runs on the same input could produce inconsistent results. For example, one run might explicitly mention "clear costophrenic angles" while another omitted this detail, and in some cases, minor discrepancies in diagnostic terminology (e.g., "mild atelectasis" vs. "minimal atelectasis") were observed. This inconsistency poses challenges for clinical applications, where reliability and reproducibility of outputs are paramount.

Third, systematic evaluation of report accuracy was constrained by project timeline limitations. While the project successfully validated the functional completeness of the report generation pipeline (e.g., generating structurally coherent, medically relevant text), it did not establish a rigorous framework to assess the *clinical correctness* of outputs. This included gaps such as not benchmarking generated reports against radiologist-annotated ground truth, not quantifying metrics for image-text alignment (e.g., whether reported findings directly correspond to visual features in the radiograph), and not collecting feedback from clinical users to evaluate real-world utility. This lack of comprehensive accuracy assessment represents a key area for refinement in subsequent iterations.

These observations collectively highlight actionable pathways for enhancing the module's robustness: scaling to larger multimodal models (e.g., LLaVA-7B) to mitigate hallucinations, tuning generation hyperparameters (e.g., lowering temperature or using beam search) to improve stability, and developing a structured evaluation pipeline with clinical input to validate report accuracy.

5.2.2 Observations and Limitations in LLaVA-7B-Related Work

The fine-tuned LLaVA-7B model showed partial yet meaningful ability in automated radiology report generation, yielding structured and clinically plausible outputs for most standard chest X-ray cases. That said, several limitations were evident: constrained by a relatively small training dataset (approximately 7,000 samples), the model occasionally misinterpreted subtle findings—particularly low-intensity opacities, small effusions, or mild cardiomegaly—and demonstrated

inconsistent performance in spatial reasoning and anatomical side distinction (left versus right). These shortcomings highlight the need for both dataset expansion and targeted enhancement in the medical domain, with several avenues for improvement emerging for future work.

To begin with, dataset scales remain a key limiting factor. With only around 7,000 high-quality samples, the model mastered strong format imitation but achieved limited semantic generalization. Expanding the dataset to at least 20,000–25,000 diverse samples would expose the model to a broader range of disease manifestations, patient demographics, and imaging artifacts, stabilizing the visual encoder's representation space and enhancing the robustness of multimodal fusion.

Additionally, domain-specific tokenization requires strengthening. Many complex medical terms—such as "costophrenic sulcus blunting", "bibasilar consolidation", and "subsegmental atelectasis"—were split into multiple subword tokens, weakening their semantic representation. Incorporating medical ontologies like RadLex and SNOMED CT into the tokenizer vocabulary, alongside retraining embeddings on medical corpora, would enable the model to treat these phrases as single, meaningful units.

Furthermore, data augmentation should extend beyond textual prompts. Techniques such as random cropping, brightness/contrast adjustments, and simulated occlusion can be applied to images to mimic real-world variability, while paraphrasing and back-translation can enrich the linguistic diversity of textual inputs; controlled uncertainty prompts could also help the model learn to express diagnostic confidence appropriately.

Finally, fine-tuning strategies may be enhanced through curriculum learning—starting with simple "single abnormality" cases and gradually progressing to complex multi-pathology images. Integrating reinforcement learning from human feedback (RLHF) with radiologist evaluations could further align model outputs with expert judgment, especially for ambiguous or borderline cases.

Taken together, these improvements have the potential to transform the current prototype into a clinically valuable assistant, capable of generating consistent, interpretable, and efficient radiographic reports.

5.2.3 Observations and Limitations in RAG-Related Work

While the RAG (Retrieval-Augmented Generation) component substantially improved factual grounding and language quality, several limitations and challenges were identified during experimentation and system deployment.

1. **Dependence on Retrieval Quality:**

The generated report's accuracy is tightly coupled with the relevance of retrieved documents. When the FAISS index returned passages with loosely related content (e.g., generic disease descriptions), the model occasionally incorporated tangential information into the report. This highlights the need for finer-grained retrieval filters, relevance re-ranking, and entity-aware query expansion.

2. **Context Window Constraints:**

The GLM-4V API imposes token limits that restrict how many retrieved passages can be injected. Longer cases with multiple abnormalities often required aggressive text summarization, occasionally leading to omitted secondary findings. Future iterations could apply adaptive context compression or hierarchical retrieval to preserve completeness.

3. **Heterogeneity of Knowledge Sources:**

Mixing structured (CheXpert) and unstructured (Wikipedia) text improved coverage but introduced

stylistic inconsistencies. Some Wikipedia segments contained non-clinical phrasing, which slightly degraded the professional tone of the generated clinician reports. Incorporating curated medical corpora (e.g., UMLS, PubMed abstracts) could enhance domain consistency.

4. Latency and Cost of API-based Retrieval-Generation:

Although asynchronous calls mitigated blocking, retrieval + generation through GLM-4V still incurred noticeable delays (5–8 s per request) and usage costs under heavy workloads. These factors limit scalability for large-scale screening deployment unless caching and batching strategies are adopted.

Summary

The RAG approach successfully demonstrated that structured retrieval can bridge the gap between language fluency and clinical reliability. However, its effectiveness depends on retrieval precision, context integration, and source quality. Addressing these issues will be essential for transitioning from research prototypes to deployable, evidence-grounded medical reporting systems.

6. Appendices

6.1 Mapped System Functionalities against knowledge, techniques and skills of modular courses: MR, RS, CGS

module	Knowledge taught	System functionality
MS/RS/ CGS	Recognize machine reasoning/ intelligent reasoning needs across different industry application	Focused on the medical field, identifying needs like AI-assisted chest radiograph screening and reducing radiologists' workload via automated reports.
MS	Learn key machine reasoning methods (e.g., rule-based logic, knowledge representation, ML, uncertainty handling)	Used ML (LLaVA-4B fine-tuning, classifiers), RAG for knowledge retrieval, and classifier probability scores to manage reasoning uncertainty.
MS	Use ML to extract industry-specific knowledge and turn business rules into computer-readable forms	Applied ML to pull medical knowledge (e.g., X-ray feature-clinical finding links) and encoded rulesvia data scripts.
MS	Compare structures and core techniques of different reasoning systems	Evaluated three report-generation setups: end-to-end LLaVA fine-tuning, classifier→generator pipeline, and API+RAG—noting tradeoffs in dependency and interpretabili
MS/CGS	Design knowledge-based reasoning software modules aligned with business goals and domain knowledge	Built modules (classifier, RAG, report generator) tailored to clinical goals, like structured disease screening and hallucination reduction

MS/RS/ CGS	Build software apps using machine reasoning skills and visual system development	Created a full-stack app: React frontend (visual interface), FastAPI backend, and integrated reasoning modules (classifier, LLaVA) for end-to-end use.
RS	Incorporate advanced AI components into reasoning systems—such as heuristic search, optimization, data mining, and machine learning algorithms—alongside system integration and programming	Integrated multiple advanced AI enablers: machine learning (LLaVA-4B fine-tuning, ResNet-based disease classification), data mining, and system integration (connecting React frontend, FastAPI backend, and Colab inference services via ngrok).
RS	Break down complex business or industry scenarios into smaller subproblems, which are addressed by combining collaborative intelligent subsystems	Decomposed the complex scenario of "chest radiograph analysis + clinical report generation" into subproblems: data formatting, model fine-tuning, disease classification, report generation, and front-end/back-end interaction—each solved by dedicated subsystems that work together.