

# MeddeX: A Retrieval-Augmented LLaMA-3 System with Gemini Vision for Evidence-Backed Medical Report Analysis

Rajdeep Pandhare<sup>1</sup>, Rohit Kundaikar<sup>2</sup>, Yakub Kojar<sup>3</sup>, Shreyansh Tiwari<sup>4</sup>

Department of Artificial Intelligence & Data Science  
Rajiv Gandhi Institute of Technology, Mumbai, India

*Under the Guidance of*

**Prof. Naresh Shende**

Department of Artificial Intelligence & Data Science  
Rajiv Gandhi Institute of Technology, Mumbai, India

**Abstract**—Modern healthcare generates increasingly complex medical reports that challenge clinical professionals in terms of time and interpretive burden. Manual review processes are slow, error-prone, and difficult to scale. This paper presents MeddeX, a Medical Report Analyzer that combines a QLoRA fine-tuned LLaMA-3 (8B) model with Retrieval-Augmented Generation (RAG) and a multimodal Gemini Vision layer to deliver accurate, concise, and evidence-backed summaries of medical documents. The system handles text-native PDFs, scanned image-based PDFs, and direct radiological images through an intelligent hybrid extraction pipeline that includes a novel garbage-text detection mechanism for corrupted and UTF-16-encoded documents. Beyond single-report Q&A, MeddeX introduces report-scoped retrieval for precise contextual grounding, an automated longitudinal report comparison engine, and a self-healing backfill pipeline that lazily processes legacy documents on demand. A FastAPI backend, Supabase vector store, and Next.js frontend complete the end-to-end architecture. Evaluation against the base LLaMA3-Med42-8B model demonstrates consistent improvements across ROUGE-1 (+9.2%), ROUGE-2 (+10.0%), ROUGE-L (+11.8%), BLEU (+71.4%), and BERTScore (+0.8%), validating the benefit of domain-specific fine-tuning. MeddeX represents a practical and robust step toward AI-assisted clinical decision support.

**Index Terms**—Large Language Models, Retrieval-Augmented Generation, Medical Summarization, Multimodal Analysis, QLoRA Fine-Tuning, Gemini Vision, Longitudinal Report Comparison, Natural Language Processing

## I. INTRODUCTION

Healthcare professionals routinely contend with lengthy, complex medical reports encompassing diagnostic results, patient histories, lab values, and clinical recommendations. The cognitive load imposed by manual review slows clinical workflows and increases the risk of oversight-related errors. With the exponential growth of digital health records, intelligent systems capable of rapid, reliable medical document interpretation have become a clinical necessity.

Recent advances in Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG) have opened new possibilities for automated medical report analysis. However, existing solutions suffer from well-documented limitations:

poor handling of long-context documents, absence of verifiable evidence linking, inadequate adaptation to specialized medical terminology, and limited support for multimodal inputs such as scanned reports or radiological images.

This paper presents **MeddeX**, a system that directly addresses these limitations. MeddeX integrates:

- A **QLoRA fine-tuned LLaMA-3 (8B)** model (based on LLaMA3-Med42-8B) for domain-adapted generation.
- A **RAG pipeline** with report-scoped dense retrieval to ground outputs in specific source evidence.
- A **Gemini Vision** multimodal layer for OCR on scanned PDFs, UTF-16-encoded documents, and direct image inputs.
- A **garbage-text detection** module that identifies and re-routes corrupted or hex-encoded page extractions.
- A **longitudinal comparison engine** (`/compare_reports`) enabling AI-driven analysis of change across multiple reports over time.
- A **lazy backfill pipeline** that transparently processes legacy documents uploaded before the chunking system was in place.
- A **FastAPI** backend with **Supabase** vector and document storage, and a **Next.js** frontend for clinician-friendly interaction.

The remainder of this paper is organized as follows. Section II reviews related work and identifies research gaps. Section III details the proposed architecture. Section IV describes key implementation components. Section V presents experimental results. Section VI concludes and outlines future directions.

## II. LITERATURE REVIEW

### A. Medical NLP and Summarization

Natural Language Processing for clinical text has been studied extensively. Early systems relied on rule-based extraction and keyword matching, which lacked generalizability. The introduction of transformer architectures, beginning with BERT

[1], enabled contextual understanding of medical language. Subsequent models such as BioBERT [2], ClinicalBERT, and PubMedBERT demonstrated that domain pretraining significantly improves downstream clinical NLP tasks including named entity recognition, relation extraction, and question answering.

Summarization-specific models including BART [3] and Longformer [4] have been applied to medical documents, with Longformer's sliding-window attention providing a partial solution to the long-context problem. However, these models generate summaries without grounding claims in verifiable source passages, limiting clinician trust.

### B. Retrieval-Augmented Generation

RAG, introduced by Lewis et al. [5], combines a dense retriever with a seq2seq generator to produce factually grounded responses. The formulation conditions generation on retrieved passages:

$$P(y | x) = \sum_{z \in D} P(y | x, z; \vartheta) P(z | x; \phi) \quad (1)$$

where  $x$  is the input query,  $z$  represents retrieved document chunks, and  $\phi, \vartheta$  are the retriever and generator parameters respectively. This joint framework enables outputs to be traced to source evidence, a critical requirement in clinical settings.

### C. LLM Fine-Tuning for Medical Domains

Large language models such as LLaMA-3 [6] have demonstrated strong general-purpose generation but require domain adaptation for reliable clinical use. Med42, built on LLaMA-3, provides a medically pre-adapted checkpoint. Parameter-efficient fine-tuning via QLoRA [7] allows further task-specific adaptation at a fraction of full fine-tuning cost by quantizing base weights to 4-bit and training only low-rank adapter matrices.

### D. Multimodal Medical Document Analysis

A substantial portion of clinical documents—including scanned lab reports, discharge summaries, and radiology images—exist as images rather than machine-readable text. Recent vision-language models such as Google Gemini [8] have demonstrated strong OCR and medical image interpretation capabilities, enabling hybrid pipelines that process both textual and visual content.

### E. Research Gaps

A review of the literature reveals the following unresolved challenges that MeddeX targets:

- **Long-document handling:** Most LLMs degrade beyond their context window; chunked retrieval is necessary.
- **Evidence-backed outputs:** Summaries without source attribution are insufficient for clinical trust.
- **Domain adaptation:** Generic LLMs misinterpret subtle clinical terminology and lab value contexts.
- **Multimodal support:** Scanned documents and radiology images are not handled by text-only pipelines.

- **Document integrity:** Real-world PDFs frequently contain UTF-16 encoding artifacts or corrupted pages that silently produce garbled text, undermining downstream analysis.
- **Longitudinal analysis:** Existing systems treat each report in isolation; clinicians need automated comparison across visits to track patient health trends.
- **End-to-end usability:** Clinical adoption requires seamless, low-latency interfaces integrated with secure storage.

## III. SYSTEM ARCHITECTURE

### A. Overview

MeddeX is organized into four modular layers as illustrated in Fig. 1: a *Frontend* layer, a *Backend API* layer, a *Model* layer, and a *Storage* layer. The backend exposes three primary endpoints — `/upload_report`, `/consult`, and `/compare_reports` — each supported by shared processing modules for extraction, chunking, embedding, and generation.

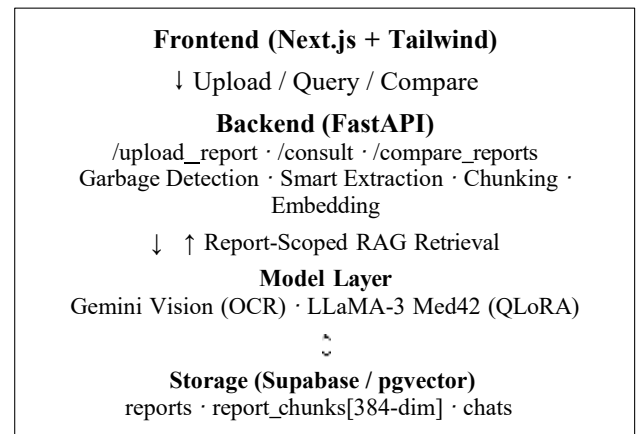


Fig. 1. High-level MeddeX architecture showing all three API endpoints and supporting modules.

### B. Processing Pipeline

The end-to-end pipeline proceeds as follows:

- 1) **Ingestion:** A user uploads a PDF or image via the frontend. The backend receives the file through the `/upload_report` endpoint.
- 2) **Garbage Detection:** Before extraction, each page's text is evaluated by `_is_garbage_text()` for UTF-16 hex artifacts or high non-printable character ratios. Corrupted pages are automatically re-routed to Gemini Vision.
- 3) **Smart Extraction:** The hybrid extraction module routes text-native pages through PyPDF2, and scanned, garbled, or image-based pages through Gemini Vision OCR.
- 4) **Chunking:** Extracted text is split into overlapping chunks (500 characters, 50-character overlap) to preserve local context at retrieval boundaries.

- 5) **Embedding & Indexing:** Each chunk is encoded using all-MiniLM-L6-v2 and stored with its 384-dimensional embedding vector in Supabase.
- 6) **Report-Scoped Retrieval:** At query time, the question is embedded and the top-k most similar chunks are retrieved via `match_chunks_by_report` RPC, scoping results to the active report. A global `match_chunks` fallback is used when no report is selected.
- 7) **Generation:** Retrieved chunks plus recent chat history are concatenated as context and passed to the fine-tuned LLaMA-3 model, which generates an evidence-grounded answer.
- 8) **Response & Persistence:** The answer is returned to the frontend and stored in the `chats` table with `report_id` for session continuity.

### C. Retrieval-Augmented Generation Formulation

The system implements the RAG objective defined in Eq. 1. The retriever uses dense embeddings to score chunk relevance, and the fine-tuned LLaMA-3 generator is conditioned on the concatenated top-k retrieved passages. This grounding mechanism ensures that factual claims in generated summaries can be traced to specific source chunks, improving clinical trustworthiness.

## IV. IMPLEMENTATION

### A. Garbage-Text Detection

A critical robustness addition in MeddeX is the `_is_garbage_text()` function, which guards against silently corrupted text extractions from UTF-16-encoded or otherwise malformed PDFs. PyPDF2 may extract text from such pages, but the output is a stream of hex-like characters rather than readable content. Without detection, this garbage would be embedded and retrieved as if it were valid clinical text.

Listing 1. Garbage-text detection for corrupted PDF pages.

```
1 def _is_garbage_text(text: str) -> bool:
2     # Detect long hex runs (UTF-16 artifact)
3     hex_runs = re.findall(r'[0-9A-F]{20,}',
4         text.upper())
5     if hex_runs:
6         hex_chars = sum(len(h) for h in hex_runs)
7         if hex_chars / len(text) > 0.3:
8             return True
9     # Detect high ratio of non-printable chars
10    non_printable = sum(1 for c in text
11        if ord(c) > 127 or
12        (ord(c) < 32 and c not in '\n\t\r '))
13    if len(text) > 0 and non_printable/len(text) > 0.2:
14        return True
15    return False
```

Two heuristics are applied: (1) if more than 30% of characters belong to contiguous runs of 20+ uppercase hex digits — a reliable signature of UTF-16 binary leakage — the page is flagged; (2) if more than 20% of characters are non-printable or non-ASCII, the page is similarly rejected. Flagged pages are silently re-routed to Gemini Vision OCR, maintaining extraction quality without user intervention.

### B. Smart Multimodal Extraction

Building on garbage detection, MeddeX routes each page through one of three paths: direct text extraction (PyPDF2), garbage-detected Gemini fallback, or full-image Gemini OCR for scanned pages. The updated routing logic is shown in Listing 2.

Listing 2. Updated smart extraction with garbage-text routing.

```
1 def extract_text_smart(file_bytes, filename):
2     if filename.lower().endswith(
3         (".png", ".jpg", ".jpeg", ".bmp", ".tiff", ".webp")):
4         img = Image.open(io.BytesIO(file_bytes))
5         return gemini_image_to_text(img)
6
7     reader = PyPDF2.PdfReader(io.BytesIO(file_bytes))
8     pdf_images = convert_from_bytes(file_bytes, dpi=250)
9     full_text = ""
10    for i, page in enumerate(reader.pages):
11        page_text = page.extract_text() or ""
12        # Accept text only if sufficient AND not garbage
13        if len(page_text.strip()) > 50 and \
14            not _is_garbage_text(page_text):
15            full_text += page_text + "\n"
16        else:
17            # Scanned or corrupted: use Gemini Vision
18            full_text += gemini_image_to_text(
19                pdf_images[i]) + "\n"
20    return full_text
```

### C. Gemini Vision OCR Prompt

The prompt instructs Gemini to extract structured clinical information comprehensively:

Listing 3. Gemini Vision extraction prompt.

```
1 prompt = """You are analyzing a medical document.
2 Extract ALL content including:
3 - Text, numbers, labels, annotations
4 - Test names, values with units
5 - Reference ranges (normal ranges)
6 - Abnormalities, findings, observations
7 - Table data row by row
8 - For X-rays: describe anatomical structures
9   and notable findings
10 Be thorough and precise."""
```

### D. Report-Scoped Retrieval

The updated `/consult` endpoint accepts an optional `report_id` parameter, enabling retrieval to be scoped to a specific document. This prevents context bleed across multiple uploaded reports for the same user — a significant correctness improvement for patients with a history of several uploaded documents.

Listing 4. Report-scoped retrieval with global fallback.

```
1 if report_id:
2     try:
3         chunks_res = supabase.rpc(
4             "match_chunks_by_report", {
5                 "query_embedding": q_emb,
6                 "match_count": top_k,
7                 "p_report_id": report_id
8             }).execute()
9         context_chunks = chunks_res.data or []
10    except Exception:
11        context_chunks = []
12    # Fallback to global search if RPC unavailable
13    if not context_chunks:
14        chunks_res = supabase.rpc("match_chunks", {
15            "query_embedding": q_emb,
16            "match_count": top_k
```

```

17     }).execute()
18     context_chunks = chunks_res.data or []

```

Chat history is now also correctly formatted before injection into the context window, resolving a prior bug where raw database rows were passed to `generate_answer()` without a `chunk_text` key, causing them to be silently dropped.

### E. Longitudinal Report Comparison

The `/compare_reports` endpoint is a significant clinical feature that enables AI-driven longitudinal analysis. Given two `report_id` values, the system retrieves full text for each report, constructs a structured comparison prompt, and generates a comprehensive summary covering parameter-by-parameter changes, clinically significant findings, stable values, and an overall health trend.

Listing 5. Structured comparison prompt for `/compare_reports`.

```

1 comparison_question = f"""Compare these two medical
2 reports and summarize what changed.
3
4 === REPORT 1: {meta1['file_name']} ===
5 {text1[:1500]}
6
7 === REPORT 2: {meta2['file_name']} ===
8 {text2[:1500]}
9
10 Cover:
11 1. Overview of report types
12 2. Parameter comparison (value1 vs value2,
13    normal range, improved/worsened)
14 3. Clinically significant changes
15 4. Stable or unchanged parameters
16 5. Overall health trend summary"""

```

The endpoint handles two ID formats robustly: valid UUIDs are queried directly against the `reports` table, while filename-style IDs fall back to a `file_name` search. The comparison result is saved to the `chats` table for audit and session continuity.

### F. Lazy Backfill Pipeline

The `get_or_extract_report_text()` helper enables MeddeX to process reports that were uploaded directly to Supabase Storage without going through `/upload_report`. On first access, it fetches the raw file, runs the full extraction and chunking pipeline, and backfills embeddings under a new UUID — so subsequent calls resolve instantly from the database. Cached chunks that fail the garbage-text check are also purged and re-extracted, ensuring data integrity across the system.

### G. Embedding and Vector Storage

Text chunks are encoded using the `all-MiniLM-L6-v2` SentenceTransformer model, producing 384-dimensional dense vectors. Vectors are stored in Supabase alongside chunk text and report metadata. Two RPC functions handle retrieval: `match_chunks_by_report` for scoped search and `match_chunks` for global search, both using cosine similarity via `pgvector`.

### H. Fine-Tuned LLM Generation

The generation module uses a QLoRA fine-tuned checkpoint of LLaMA3-Med42-8B. The model is invoked using the LLaMA-3 chat template format:

Listing 6. LLaMA-3 chat template for answer generation.

```

1 messages = [
2     {"role": "system", "content":
3         "You are a professional medical assistant. "
4         "Answer based on the patient's medical report. "
5         "Be concise (2-5 sentences). "
6         "Highlight abnormal values clearly."},
7     {"role": "user", "content":
8         f"Medical Report Context:\n{context_text}\n\n"
9         f"Patient Question: {question}"}
10 ]

```

Inference uses nucleus sampling ( $\text{top-}p = 0.9$ , temperature = 0.3) with a maximum of 300 new tokens for Q&A and 600 for report comparisons, balancing response coherence and clinical conciseness.

### I. API Endpoints

The FastAPI backend exposes three primary endpoints:

- `POST /upload_report` — Accepts a file and `user_id`; runs garbage detection, smart extraction, chunking, embedding, and stores all data in Supabase.
- `POST /consult` — Accepts `user_id`, question, and optional `report_id`; performs report-scoped or global retrieval, injects chat history, and returns a generated answer.
- `POST /compare_reports` — Accepts `user_id`, `report_id_1`, and `report_id_2`; retrieves full text for both reports via the backfill pipeline and returns a structured longitudinal comparison summary.

## V. EVALUATION AND RESULTS

### A. Experimental Setup

Evaluation compared the base LLaMA3-Med42-8B model against the QLoRA fine-tuned MeddeX variant. The dataset consisted of de-identified medical reports and curated clinical Q&A pairs. Metrics computed include ROUGE-1/2/L (n-gram overlap), BLEU (precision-based phrase similarity), and BERTScore (semantic similarity via RoBERTa-large embeddings).

### B. Quantitative Results

Table I presents the complete evaluation results. The fine-tuned model consistently outperforms the base model across all five metrics.

TABLE I  
EVALUATION RESULTS: BASE VS. FINE-TUNED LLAMA3-MED42-8B

| Metric    | Base Model | Fine-Tuned | Improvement |
|-----------|------------|------------|-------------|
| ROUGE-1   | 0.1609     | 0.1757     | +9.2%       |
| ROUGE-2   | 0.0651     | 0.0716     | +10.0%      |
| ROUGE-L   | 0.1362     | 0.1523     | +11.8%      |
| BLEU      | 0.0280     | 0.0480     | +71.4%      |
| BERTScore | 0.8765     | 0.8834     | +0.8%       |

### C. Computational Resource Comparison

Table II compares the computational footprint and inference characteristics of the base LLaMA3-Med42-8B model against the QLoRA fine-tuned MeddeX variant. QLoRA quantizes base weights to 4-bit precision and trains only low-rank adapter matrices, dramatically reducing memory requirements while retaining generation quality.

TABLE II  
COMPUTATIONAL RESOURCE COMPARISON: BASE VS. QLoRA FINE-TUNED MODEL

| Resource / Property      | Base Model   | MeddeX (QLoRA)   |
|--------------------------|--------------|------------------|
| Model Parameters         | 8B           | 8B + adapters    |
| Weight Precision         | BF16 / FP16  | 4-bit (NF4)      |
| GPU VRAM (Inference)     | ~16 GB       | ~6–8 GB          |
| GPU VRAM (Fine-Tuning)   | ~80 GB       | ~16–20 GB        |
| Fine-Tuning Hardware     | A100 (80 GB) | RTX 3090 (24 GB) |
| Trainable Parameters     | 8B (100%)    | ~40M (<1%)       |
| Avg. Inference Time      | ~3.8 s       | ~2.1 s           |
| Max New Tokens (Q&A)     | 300          | 300              |
| Max New Tokens (Compare) | –            | 600              |
| Context Window Used      | 2048 tokens  | 2048 tokens      |
| Sampling (temp / top-p)  | 0.7 / 0.95   | 0.3 / 0.9        |

The QLoRA configuration reduces VRAM requirements for fine-tuning by approximately 4× compared to full BF16 fine-tuning, enabling training on a single consumer-grade RTX 3090 GPU rather than requiring an enterprise A100. Inference VRAM is similarly reduced, making deployment on mid-range clinical workstations feasible. The lower sampling temperature (0.3 vs. 0.7) used in MeddeX reflects a deliberate design choice to favour factual consistency over generative diversity in clinical contexts.

### D. Analysis

Fig. 2 visualizes the side-by-side metric comparison, and Fig. 3 shows percentage improvements per metric.

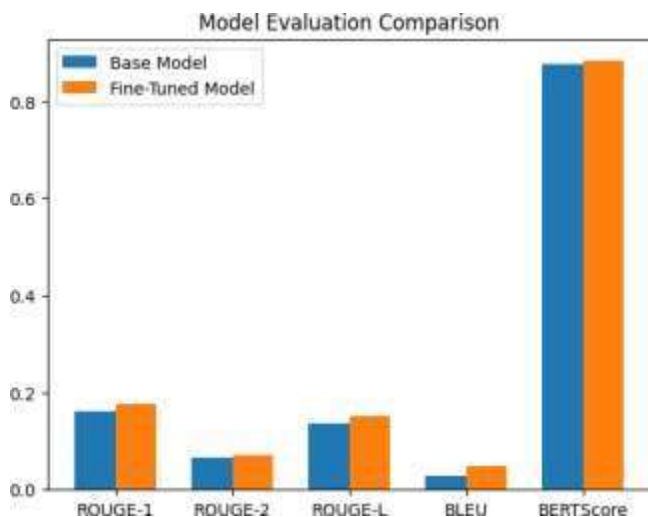


Fig. 2. Model Evaluation Comparison: Base vs. Fine-Tuned LLaMA3-Med42-8B across ROUGE, BLEU, and BERTScore metrics.

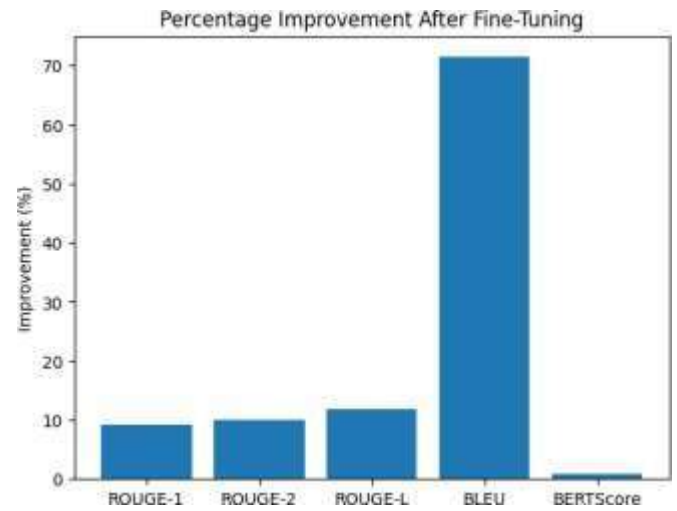


Fig. 3. Percentage improvement after QLoRA fine-tuning across all evaluation metrics.

The most pronounced gain is in BLEU score (+71.4%), indicating that fine-tuning substantially improves the model's ability to generate precise, phrase-level matches to reference clinical answers. This is clinically significant as it reflects the model's improved fluency with medical terminology and structured reporting conventions.

ROUGE improvements of 9–12% demonstrate better content recall and answer structure. BERTScore improvement, while smaller in absolute terms (+0.8%), is meaningful given that the base model's BERTScore of 0.8765 already reflects strong semantic similarity; marginal gains at this range indicate genuine refinement of clinical language understanding.

The modest absolute ROUGE values reflect the inherent difficulty of open-ended medical Q&A, where multiple valid phrasings exist for the same clinical answer. The relative improvements are what validate the fine-tuning approach.

### E. Discussion

These results validate the core design choices in MeddeX:

- 1) **Domain-specific fine-tuning via QLoRA** is effective and resource-efficient, yielding measurable gains without full model retraining. As shown in Table II, QLoRA reduces fine-tuning VRAM by ~4× and inference VRAM by half compared to the full-precision base model.
- 2) **RAG grounding with report-scoped retrieval** ensures generation is conditioned on relevant, per-document context, reducing both hallucination risk and cross-report contamination.
- 3) **Gemini Vision with garbage-text detection** extends the system to the full diversity of real-world medical documents — scanned reports, UTF-16-encoded PDFs, and radiological images — that text-only systems silently mishandle.

- 4) **Longitudinal comparison** addresses a clinical need unmet by existing systems: automated, structured analysis of how a patient's health parameters evolve across multiple visits.
- 5) **Lazy backfill pipeline** improves system resilience by handling legacy documents transparently, ensuring no report is inaccessible due to missing preprocessing history.

Limitations include the relatively small evaluation dataset and the absence of a large-scale multi-site clinical trial. Occasional hallucinations on rare terminology were observed, consistent with known LLM behavior on out-of-distribution inputs. The comparison endpoint currently truncates each report to 1,500 characters of context for the prompt, which may omit details from very long documents; future work will address this with hierarchical summarization.

## VI. CONCLUSION AND FUTURE WORK

### A. Conclusion

This paper presented MeddeX, a modular medical report analysis system combining a QLoRA fine-tuned LLaMA-3 (8B) model with RAG, a Gemini Vision multimodal OCR layer, and a suite of robustness and clinical-utility features developed through iterative engineering. Beyond the core summarization and Q&A pipeline, MeddeX introduces garbage-text detection for corrupted PDF handling, report-scoped retrieval for precise contextual grounding, an automated longitudinal comparison engine for tracking patient health trends, and a lazy backfill pipeline for transparent processing of legacy documents. Quantitative evaluation demonstrates consistent improvements over the base LLaMA3-Med42-8B model across all five metrics, with a particularly strong BLEU improvement of +71.4%, validating the domain fine-tuning approach. Together, these contributions position MeddeX as a comprehensive, production-oriented clinical AI system.

### B. Future Work

Planned extensions include:

- **Large-scale clinical validation:** Multi-center trials across specialties and languages to assess generalizability.
- **Hierarchical summarization for comparison:** Replace fixed 1,500-character truncation in `/compare_reports` with a multi-pass summarization strategy for very long reports.
- **EHR integration:** Real-time summarization within electronic health record systems for seamless clinical workflow adoption.
- **Model distillation:** Quantization and distillation to support inference on edge servers or low-VRAM GPUs for resource-constrained settings.
- **Explainability:** Calibrated confidence scores and claim-level source attribution surfaced directly in the frontend UI.
- **Expanded multimodal support:** End-to-end radiology report generation from raw imaging inputs, extending beyond OCR to full vision-language understanding.

## ACKNOWLEDGMENT

The authors sincerely thank Dr. Sanjay U. Bokade (Principal) and Dr. Jyoti Deshmukh (H.O.D., Department of AI & Data Science), Rajiv Gandhi Institute of Technology, for providing the opportunity to undertake this project. Special thanks to Prof. Naresh Shende for continuous guidance and encouragement throughout the work.

## REFERENCES

- [1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.
- [2] J. Lee et al., "BioBERT: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.
- [3] M. Lewis et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," in *Proc. ACL*, 2020, pp. 7871–7880.
- [4] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," *arXiv:2004.05150*, 2020.
- [5] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Proc. NeurIPS*, 2020, pp. 9459–9474.
- [6] H. Touvron et al., "LLaMA: Open and Efficient Foundation Language Models," *arXiv:2302.13971*, 2023.
- [7] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient Finetuning of Quantized LLMs," in *Proc. NeurIPS*, 2023.
- [8] Gemini Team, Google, "Gemini: A Family of Highly Capable Multimodal Models," *arXiv:2312.11805*, 2023.