

**ENHANCING MEDICAL QUESTION-ANSWERING THROUGH LOCAL
FINE-TUNING OF SMALL LANGUAGE MODELS**

A DISSERTATION

**SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE AWARD OF THE DEGREE
OF**

**MASTERS OF TECHNOLOGY
IN
ARTIFICIAL INTELLIGENCE**

Submitted By:

ARUNAV PRAKASH

2K23/AFI/28

Under the supervision of

PROFESSOR SHAILENDER KUMAR



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering) Bawana Road, Delhi-110042

May, 2025

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College Of
engineering) Bawana Road,
Delhi-110042

DECLARATION BY THE CANDIDATE

I, Arunav Prakash, Roll No. 2K23/AFI/28, a student of Masters of Technology (Artificial Intelligence), hereby declare that the Project Dissertation titled “*ENHANCING MEDICAL QUESTION-ANSWERING THROUGH LOCAL FINE-TUNING OF SMALL LANGUAGE MODELS*”, submitted by me at the Department of Computer Science & Engineering, Delhi Technological University, Delhi, in partial fulfillment of the requirements for the award of the Master of Technology degree, is my original work. Any references or sources used have been appropriately cited. This work has not been submitted previously for the award of any degree, diploma, associateship, fellowship, or similar title or recognition anywhere else.

Place: DTU, Delhi

Arunav Prakash
(2K23/AFI/28)

Date:

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College Of
engineering) Bawana Road,
Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled “**ENHANCING MEDICAL QUESTION-ANSWERING THROUGH LOCAL FINE-TUNING OF SMALL LANGUAGE MODELS**”, submitted by Arunav Prakash, Roll No. 2K23/AFI/28, of the Department of Computer Science & Engineering, Delhi Technological University, Delhi, in partial fulfillment of the requirements for the award of the degree of Master of Technology (Computer Science & Engineering), is a record of the project work carried out by the student under my supervision. To the best of my knowledge, this work has not been submitted, either in part or in full, for any degree or diploma at this University or elsewhere.

Place: DTU, Delhi

Prof. Shailendra Kumar

Date:

SUPERVISOR

ABSTRACT

This thesis investigates how parameter-efficient fine-tuning techniques can bring powerful language models into everyday clinical environments without sacrificing performance or patient privacy. We begin by selecting a representative subset (16,412 pairs) of the MedQuAD medical question-answer dataset and adapt two LLaMA variants, a 3 billion-parameter model using LoRA adapters and an 8 billion-parameter model with 4-bit QLoRA quantization, entirely on a single RTX 4060 GPU with 8 GB VRAM. Training and inference both complete in under five hours, demonstrating that consumer-grade hardware can support domain-specific LLMs when only lightweight adapters are updated.

To evaluate model outputs, we develop a multi-axis scoring framework, relevance, accuracy, conciseness, and completeness, automatically produced by a locally hosted LLaMA 3.1 8B judge via Ollama. This structured, human-aligned approach reveals clear differences: the 8 B QLoRA model consistently outperforms its 3 B counterpart across all four dimensions (mean score 7.16 vs. 6.96), while traditional overlap metrics like ROUGE fail to capture these gains. We show that ROUGE’s reliance on n-gram matching penalizes valid paraphrases and richer contextual detail, making it an unreliable proxy for clinical language quality.

Our contributions include a reproducible, on-device pipeline for fine-tuning and evaluation, compelling evidence that aggressive quantization need not compromise model expressivity, and a practical blueprint for deploying privacy-preserving medical chatbots in resource-constrained settings. We conclude by outlining future directions, ensemble judging, expanded empathy and readability metrics, dynamic adapter libraries, and hybrid human-in-the-loop workflows, to further bridge the gap between scalable automation and clinical safety.

ACKNOWLEDGEMENT

I am extremely grateful to Prof. Shailendra Kumar, Department of Computer Science Engineering, Delhi Technological University, Delhi for providing invaluable guidance and being a constant source of inspiration throughout my research. I will always be indebted to him for the extensive support and encouragement he provided.

Arunav Prakash

Roll-No. 2K23/AFI/28

LIST OF TABLES

1. Table 1 : RAG vs Fine-tuning
2. Table 1 : VRAM Requirements for Different Models
3. Table 2 : Fine-tuned Model Performance across different dimensions

LIST OF FIGURES

1. Fig 1 : LSTM
2. Fig 2 : Transformer Architecture
3. Fig 3 : RAG Workflow
4. Fig 1 : Raw Dataset Details
5. Fig 2 : Fine-tuning workflow
6. Fig 3 : JSON Schema Definition

Table of Contents

DECLARATION BY THE CANDIDATE	I
CERTIFICATE	II
ABSTRACT	III
ACKNOWLEDGEMENT	IV
LIST OF TABLES	V
LIST OF FIGURES	VI
1.INTRODUCTION	1
1.1 Background and Motivation	1
1.2 Problem Statement	1
1.3 Objectives	2
1.4 Contributions	2
1.5 Structure of the Thesis	3
2. LITERATURE REVIEW	4
2.1 Large Language Models in Healthcare	4
2.2 Brief Evolution of NLP Architectures and Adaptation Methods	4
2.3 Domain Adaptation: Fine-Tuning vs. RAG	6
2.3 Challenges in Fine-Tuning Large Models	8
2.4 Parameter-Efficient Fine-Tuning (PEFT): LoRA and QLoRA	9
2.5 Evaluation Metrics for Language Models and Challenges	10
3. METHODOLOGY AND IMPLEMENTATION	12
3.1 Dataset Description (MedQuAD) and Data Preprocessing	12
3.2 Model Selection and Justification	12
3.3 Fine-Tuning Strategies	13
3.4 Implementation of LoRA	14
3.5 Implementation of Quantized LoRA (QLoRA)	15
3.6 Training Configuration and Workflow	16
4. EVALUATION DESIGN	18
4.1 Evaluation Objectives	18
4.2 Evaluation Metrics	19
4.3 Automated Scoring Pipeline & Tooling	20
4.4 Limitations and Trade-offs	22
5. RESULT AND ANALYSIS	24
5.1 Model Performance Across Dimensions	24
5.2 Inadequacy of ROUGE for Medical QA and Rationale for LLM as Judge	25
6. DISCUSSIONS	27
6.1 Interpretation of Results	27
6.2 Advantages of PEFT in Resource-Constrained Environments	28
6.3 Limitations and Challenges	28
7. CONCLUSION AND FUTURE WORK	30

7.1 Summary of Findings	30
7.2 Contributions to the Field	31
7.3 Recommendations for Future Research	31
REFERENCES	33

1.INTRODUCTION

1.1 Background and Motivation

In recent years, large language models (LLMs) have become increasingly central to natural language processing tasks, enabling systems to interpret and generate human-like text across a wide range of domains. With methods such as fine-tuning and retrieval-augmented generation (RAG), it is now possible to tailor these models to perform specific tasks without starting from scratch. This shift makes it possible to consider applications in fields like medical science, where the stakes are high and the need for accurate language understanding is critical.

However, deploying LLMs in real-world medical contexts is far from straightforward. One major challenge lies in the practical resources required, full-scale fine-tuning of large models demands powerful hardware and access to large, curated datasets, which are not always available to smaller research teams or institutions. On top of that, privacy remains a serious concern. Medical data is inherently sensitive, and uploading it to external servers or cloud-based APIs raises questions about confidentiality and compliance with healthcare regulations.

These limitations make it necessary to explore more efficient ways to adapt language models for medical use, approaches that reduce the computational burden while keeping everything local and secure. Instead of depending on full fine-tuning or third-party APIs, it makes more sense to look into parameter-efficient techniques that can modify only the essential parts of a model. This way, it's possible to achieve useful domain adaptation without compromising privacy or hitting hardware limitations.

In this work, the focus is on building a medical question-answering system by fine-tuning an existing LLM using lightweight methods that preserve both performance and practicality. The idea is not to reinvent the wheel, but to work smarter with what's already available, making powerful language models work in settings where resources are limited but the need for accurate information is crucial.

1.2 Problem Statement

Bringing large language models (LLMs) into real-world healthcare applications is far from straightforward. Two problems stand out right away. First, the full fine-tuning of these models demands serious computing power, GPUs with large memory, long training durations, and constant energy use. Not every hospital or research unit has that kind of setup. In fact, most don't. Second, and just as important, is the question of data privacy. Medical data is sensitive. It's not just about ethics; there are

strict rules, such as HIPAA, that demand tight control over who sees what, and how patient information is used.

Traditional training approaches often require uploading data to cloud platforms for processing or inference. That's convenient, sure, but not always acceptable when private health records are involved. Even with anonymization, the risks aren't zero. Many institutions would rather avoid any chance of a privacy breach entirely.

This situation makes it clear that relying on conventional fine-tuning methods isn't a good fit for healthcare. We need a different approach, one that can be done locally, without shipping sensitive data offsite, and without needing access to clusters of high-end hardware. That's where parameter-efficient fine-tuning methods come in. These allow LLMs to be adapted to medical tasks with a fraction of the resources, often by training only a small number of parameters instead of updating the whole model.

Solving these challenges isn't just a technical hurdle, it's essential if we want LLMs to have any meaningful role in real clinical environments. If the models can't be trusted to handle patient data securely, or if they're too expensive to run, they simply won't be used. The focus, then, should be on building systems that respect both the privacy of the patient and the resource constraints of the organization. Only then can these powerful models find a real place in modern healthcare.

1.3 Objectives

The primary objectives of this thesis are listed below :

1. *To investigate and implement parameter-efficient fine-tuning techniques, specifically Low-Rank Adaptation (LoRA) and Quantized LoRA (QLoRA), for adapting LLMs to the medical domain.*
2. *To develop a privacy-preserving medical question-answering system by fine-tuning a LLaMA 3.1 8B model using the MedQuAD dataset within a local computing environment.*
3. *To evaluate the performance of the fine-tuned model using metrics such as relevance, accuracy, conciseness, and completeness, ensuring its efficacy in providing medical information.*
4. *To compare the effectiveness of LoRA and QLoRA techniques in terms of model performance and resource utilization, providing insights into their suitability for healthcare applications.*

1.4 Contributions

This thesis makes the following contributions :

- Demonstrate the feasibility of fine-tuning Small Language Models (SLMs) for medical

applications using parameter-efficient techniques within resource-constrained, local environments.

- Present a comprehensive evaluation of LoRA and QLoRA methods, highlighting their impact on model performance and computational efficiency in the context of medical question-answering systems with same resource constraints.
- Develop a medical chatbot capable of giving accurate and concise information, aligning with data privacy regulations and addressing the specific needs of the healthcare providers.
- Provide a framework for assessing the quality of responses generated by the model using domain-relevant metrics, contributing to the broader field of LLM evaluation methodologies.

1.5 Structure of the Thesis

The thesis is organized as follows:

1. Chapter 1: Introduces the research background, motivation, problem statement, objectives, contributions, and outlines the thesis structure.
2. Chapter 2: Reviews related literature on LLMs in healthcare, challenges in fine-tuning large models, parameter-efficient techniques, and evaluation metrics.
3. Chapter 3: Details the methodology, including dataset description, data preprocessing, model selection, fine-tuning strategies, and training configurations.
4. Chapter 4: Describes the implementation of the system, covering the integration of fine-tuning techniques, local deployment, and inference processes.
5. Chapter 5: Presents the evaluation design, outlining the objectives, metrics, automated scoring pipeline, and discusses limitations and trade-offs.
6. Chapter 6: Analyzes the results, comparing the performance of LoRA and QLoRA, and discusses the implications of the findings.
7. Chapter 7: Concludes the thesis, summarizing key findings, contributions, and suggests directions for future research.

2. LITERATURE REVIEW

2.1 Large Language Models in Healthcare

In recent years, models like GPT-4, BERT, and their successors have completely changed how we think about language understanding. They're capable of picking up patterns, generating nuanced text, and adapting across tasks that were once considered too complex for machines. Healthcare, being a field that deals heavily with information, much of it unstructured, is starting to see the real-world value of these models.

The possibilities are wide-ranging. Clinicians can use LLMs to help sift through dense medical literature faster, generate summaries of patient histories, or even produce initial drafts of discharge notes. For patients, these models can simplify complex medical language, making diagnoses and treatments easier to grasp. This kind of "bridge" between technical medical information and human understanding could be huge for improving doctor-patient communication.

But, of course, nothing is that straightforward. Medical data isn't just another text corpus, it's private, sensitive, and heavily regulated. Using cloud-based LLMs introduces concerns about patient privacy and compliance with frameworks like HIPAA. And even when privacy is addressed, there's still the issue of trust. An LLM can't afford to "guess" when generating responses in a medical setting; lives are at stake.

That's where fine-tuning comes in. Generic models can only go so far. For healthcare use, LLMs need domain-specific training so they can speak the language of medicine, recognize context, and avoid making dangerous assumptions. Even then, their outputs should be interpretable, something that can be validated and explained if needed.

So, while the potential is there, integrating LLMs into clinical workflows isn't just a technical task. It's about aligning accuracy, ethics, and privacy with the real-world messiness of medical practice. [1]

2.2 Brief Evolution of NLP Architectures and Adaptation Methods

Early attempts to model natural language with neural networks relied heavily on Recurrent Neural Networks (RNNs). In an RNN, each input token is processed in sequence, with a hidden state carrying information forward in time. While this structure captures short-term dependencies reasonably well, it struggles to remember information over longer spans, a problem known as the vanishing gradient. To address this, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures

introduced gated mechanisms that regulate the flow of information, allowing models to retain context over hundreds of time steps. These advances made it practical to apply neural models to tasks like machine translation and speech recognition.

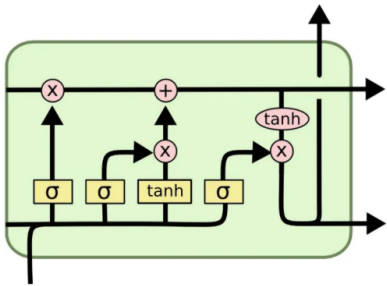


Fig 1 : LSTM

However, sequential processing inherently limits parallel computation. In 2017, Vaswani et al. proposed the Transformer, which replaced recurrence with a self-attention mechanism. Instead of stepping through tokens one by one, the Transformer computes pairwise “attention scores” between every token in the input at once. This enables the model to weigh the relevance of each token relative to every other token, dynamically focusing on the most important relationships—whether two words are adjacent or hundreds of positions apart. Crucially, self-attention operations can be parallelized across modern hardware, allowing Transformers to be trained on massive text corpora in a fraction of the time required by RNN-based models.[15]

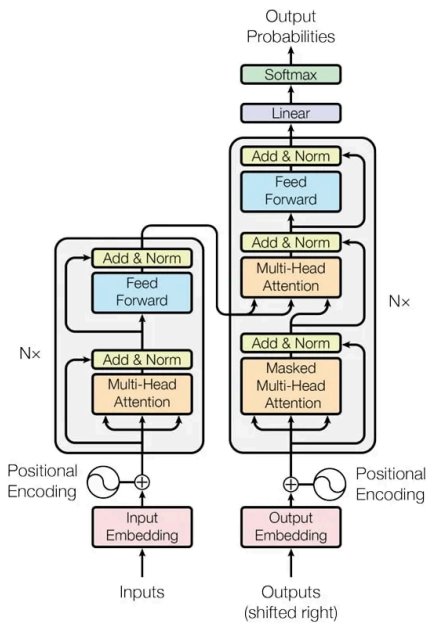


Fig 2 : Transformer Architecture

The result was a sea change in natural language processing. Pre-trained Transformer models such as BERT, GPT-2, and their successors demonstrated that a single, large model fine-tuned on specific tasks could outperform bespoke architectures. More recent “foundation models” like LLaMA and GPT-3 scale this idea to tens or hundreds of billions of parameters, learning broadly useful linguistic patterns and world knowledge during pre-training and then adapting to downstream tasks with relatively little additional data. In essence, the field has moved from carefully engineered sequence models toward a unified Transformer backbone that can be repurposed almost universally, delivering faster training times, better performance, and straightforward transfer to new tasks.

2.3 Domain Adaptation: Fine-Tuning vs. RAG

Once a Transformer is pre-trained, the next challenge is to specialize it for a particular domain or task. The most straightforward method—full fine-tuning—updates all model parameters on task-specific data. This approach often yields strong task performance but comes at a high computational cost and can lead to “catastrophic forgetting,” where the model’s general language abilities degrade as it focuses on niche data. In regulated or resource-constrained environments, full fine-tuning may be impractical or even impermissible when data privacy or hardware limitations are a concern

When designing systems for knowledge-intensive tasks, two popular approaches often emerge: retrieval-augmented generation (RAG) and fine-tuning. Both methods leverage pre-trained language models but differ fundamentally in their strategies to incorporate domain-specific knowledge. This section explores the distinctions between these approaches, highlighting their strengths, limitations, and ideal use cases to provide a clearer understanding of when to adopt each method.

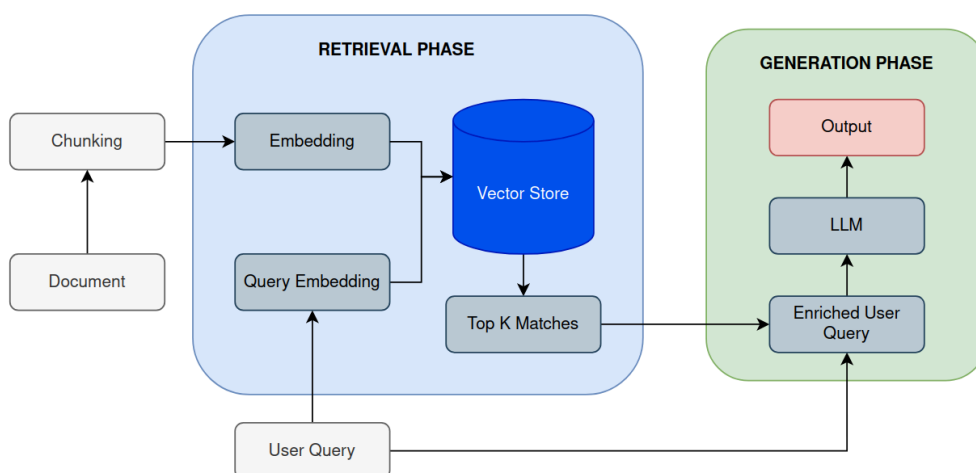


Figure 3 : RAG Workflow

Aspect	RAG Systems	Fine-Tuning
<i>Core Concept</i>	Combines retrieval of external data with a pre-trained language model to generate contextually rich answers.	Modifies the weights of a pre-trained model by training it on a specific dataset to specialize its responses.
<i>Flexibility</i>	Highly flexible as the external knowledge base can be updated without retraining the model.	Static once trained; any update to the knowledge requires retraining the model.
<i>Data Requirements</i>	Requires a large, well-structured knowledge base for retrieval but less labeled data for the model itself.	Requires a significant amount of labeled data specific to the target domain for effective fine-tuning.
<i>Scalability</i>	Easily scalable to new domains by updating the knowledge base.	Limited scalability as fine-tuning for new domains demands new datasets and retraining cycles.
<i>Cost and Efficiency</i>	Lower computational cost for updates since retrieval mechanisms are independent of the model.	High computational cost for retraining, especially for large models like GPT or LLaMA.
<i>Performance on Domain-Specific Queries</i>	Highly effective if the knowledge base is comprehensive and retrieval is accurate.	Generally better performance when trained on high-quality, domain-specific data, but low flexibility.
<i>Response Contextuality</i>	Responses are grounded in the most relevant retrieved data, ensuring up-to-date and precise answers.	Responses are limited to the knowledge captured during training, which may become outdated.
<i>Maintenance</i>	Easier to maintain by regularly updating the knowledge base.	Requires periodic retraining to incorporate new knowledge or rectify errors.
<i>Adaptation Speed</i>	Can adapt to new knowledge	Requires significant time for retraining

	instantly through retrieval updates.	and testing to adapt to new knowledge.
<i>Hallucination Risk</i>	May hallucinate if retrieval fails or irrelevant information is retrieved.	May hallucinate if the training data is incomplete or biased, without external grounding to correct it.

Table 1 : RAG vs Fine-tuning

2.3 Challenges in Fine-Tuning Large Models

When we use large language models for specialized tasks, we run into several practical issues. below are the main ones,:

1. *Expensive Hardware* : We need powerful GPUs with huge memory to fine-tune full sized LLMs. Renting or purchasing this hardware can cost thousands of dollars. Training may take days or weeks, and every hour on the GPU adds to the bill.
2. *Risk of Overfitting* : We often have limited labeled data for specialized domains like medical science. If we train too long on that small dataset, the model memorizes (overfits) patterns instead of learning generalizable insights. As a result, it performs well on our training questions but stumbles on anything new.
3. *Catastrophic Forgetting* : We begin with a model that understands broad language patterns. During the fine-tuning process, if we update all the parameters, we erase some of the general knowledge. Later, the model may respond accurately on our medical prompts but lose fluency or context in everyday language.
4. *Sensitive Data Handling* : In healthcare, the data we use to train must remain private. Uploading patient records or clinical notes to a third party can violate regulations and maybe lessen patient trust. We must ensure all fine-tuning happens locally and behind secure firewalls.
5. *Long Training Timelines* : Because each training step processes huge batches of tokens, many hours are spent on just on a single epoch. With multiple experiments, and hyper-parameter tuning, the total time adds up quickly and slows down research progress.
6. *Complex Hyper-parameter Tuning* : We need to experiment with learning rates, batch sizes, optimizer types, dropout settings and more. Each configuration demands a full on training run to validate, and the combinatorial space can be overwhelming without automation.

7. *Infrastructure and Maintenance* : We must manage versions, dependencies, and data pipelines. Even minor library upgrades can break our training scripts. Maintaining a stable environment consumes engineering time that could otherwise go into model improvements.

Because of these challenges, we recognize the need for methods that allow us to fine-tune models efficiently, securely, and with minimal hardware, while still achieving high accuracy in specialized domains.

2.4 Parameter-Efficient Fine-Tuning (PEFT): LoRA and QLoRA

Parameter-Efficient Fine-Tuning, or PEFT, has emerged as a practical way for us to tailor massive pre-trained language models to new tasks without overhauling every single weight. Rather than retraining billions of parameters from scratch, we selectively adjust only a small fraction, typically the layers or subcomponents most critical for the new domain. By focusing our updates on these targeted parameters, we dramatically cut down the compute time, memory footprint, and energy consumption that full fine-tuning would demand.

In practice, we begin with a model that already “knows” general language patterns, and we introduce compact, trainable modules, often in the form of low-rank matrices, that capture domain-specific information. During training, only the new parameters are optimized, while most of the original network remains frozen. This approach lets us retain the model’s broad linguistic knowledge, its ability to handle grammar, world facts, and common sense, while incorporating new information that steer it toward medical terminology, clinical phrasing, or any specialized jargon we care about.

From our perspective, the most compelling advantage of PEFT is two-fold. First, we can run experiments on a single workstation equipped with a modest GPU, rather than needing access to a multi-GPU cluster. Second, because we are not overwriting the core weights, we avoid catastrophic forgetting: the model keeps its general capabilities intact even as it learns new, domain-specific tasks. Overall, PEFT methods strike a careful balance between efficiency and performance, making them ideal for research teams and institutions that must respect both resource limits and the need for high-quality, domain-adapted language understanding. [2][8][10]

We concentrate on two leading parameter-efficient strategies that have proven their worth in real-world applications:

- **Low-Rank Adaptation (LoRA)** : We start by embedding small, trainable matrices within the existing layers of a Transformer model. Rather than altering the original weight tensors, LoRA

lets us insert paired matrices whose product approximates the desired adjustment. During fine-tuning, we update only these new low-rank components, while the vast majority of the network remains untouched. By doing so, we shrink the number of parameters we need to optimize by orders of magnitude. This direct targeting of model subspaces translates into dramatic reductions in GPU memory usage and slashes training time, often turning multi-day runs into experiments that finish within hours on a single workstation.[9]

- **Quantized LoRA (QLoRA)** : Building on LoRA's efficiency, we apply weight quantization to compress the model even further. QLoRA represents its parameters in lower bit-width formats, four-bit or eight-bit integers, while still preserving the fine-tuned low-rank matrices in higher precision. This hybrid scheme lets us fit large models into the memory limits of more modest GPUs, without sacrificing the domain-specific gains that LoRA provides. As a result, we maintain nearly identical performance on downstream tasks, even when working with quantized backbones that demand just a fraction of the compute and storage of their full-precision counterparts.[11][13]

Together, LoRA and QLoRA form a powerful toolkit for adapting large language models to specialized fields such as healthcare. They allow us to balance the need for rigorous domain adaptation, strict data privacy, and the practical constraints of on-premises hardware.

2.5 Evaluation Metrics for Language Models and Challenges

Evaluating the performance of LLMs requires measuring both quantitative and qualitative metrics to assess various aspects of generated text. Key evaluation metrics include : [4]

1. *BLEU (Bilingual Evaluation Understudy)*: The overlap between machine-generated text and reference translation is assessed, popularly used for rating machine translation tasks.
2. *ROUGE (Recall-Oriented Understudy for Gisting Evaluation)*: The quality of summaries measured by the overlap of n-grams between the generated and reference texts. [3]
3. *Accuracy and F1 Score*: Used to measure the correctness of the model's predictions for classification tasks .
4. *Human Evaluation*: Involves human judges assessing the relevance, coherence, and fluency of the model's outputs, providing insights into the model's practical utility.

Challenges

There are several challenges with traditional metrics of evaluating LLM responses for example the ROUGE method or BLEU make use of overlaps between generated response and reference texts, now in theory if “exact” responses are what we seek then this would be a good way to measure the output of LLMs, but in practice it is not the case when dealing with specialized domains like medicine, many general concepts could be explained in great details or many specific concepts could be explained in too simple terms which poses the risk of actual information losses in critical settings.

To overcome the aforementioned issue, we have a tedious yet feasible solution, bring in actual human judges who provide human feedback, as done by top LLM providers[14], but this is a good solution when dealing with base model, and not fine-tuning, for fine-tuning engaging experts from critical domain like medicine is not a sustainable solution.

Now this thesis work emphasizes small and efficient language models that run on consumer grade systems, so an obvious solution to the above issue is to use a more powerful LLM as a judge.

3. METHODOLOGY AND IMPLEMENTATION

3.1 Dataset Description (MedQuAD) and Data Preprocessing

The Medical Question Answering Dataset (MedQuAD) is a comprehensive collection of medical information comprising 47,457 question-answer (QnA) pairs derived from 12 authoritative websites maintained by the U.S. National Institutes of Health (NIH). These sources include platforms such as Cancer.gov, MedlinePlus, and the Genetic and Rare Diseases Information Center (GARD). The dataset comprises of 37 distinct question types, covering areas like treatment, diagnosis, side effects, etcetera, associated with diseases, drugs, and other medical entities.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16412 entries, 0 to 16411
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   question    16412 non-null  object
1   answer      16407 non-null  object
2   source      16412 non-null  object
3   focus_area  16398 non-null  object
dtypes: object(4)
memory usage: 513.0+ KB
```

Fig 4 : Raw Dataset Details

Each QA pair in MedQuAD is provided with metadata, including:

- Question Type: Type of question (e.g., Treatment, Diagnosis).
- Question Focus: Area of focus of the question.
- Source : The official source of the information.

For the purposes of this study, a subset of 16,412 QA pairs was extracted from the original MedQuAD dataset. The dataset was randomized before the extraction.

Licensing Statement : The MedQuAD dataset is publicly available and distributed under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original authors and source are credited.

For the Data Preprocessing step, the null values were removed and only Questions and Answers columns were retained.

3.2 Model Selection and Justification

Unsloth [7] is a popular python library designed to significantly speed up and optimize the fine-tuning process and is easily reproducible, it makes use of PEFT techniques like LoRA and QLoRA and has been specifically designed to work under resource constrained systems or consumer grade systems. The

system available had 8 GB of VRAM so the model chosen were as below :

1. LoRA : Llama 3.2 - 3 billion parameters, Instruction Fine-tuned (unsloth/Llama-3.2-3B-Instruct)
2. LoRA : Llama 3.1 - 8 billion parameters, Instruction Fine-tuned, Bits and Bytes Quantized for 4 Bit (unsloth/Meta-Llama-3.1-8B-Instruct-bnb-4bit)

Unsloth makes use of PEFT (Parameter-Efficient Fine-Tuning) adapters, which are small neural network modules inserted into pre-trained models to minimize computation and speed up training. They make sure that only a small part of the entire neural network is updated during the training process.

The model selection was made based on two factors, firstly instruct models are better suited for use cases where there is a certain set of instructions to be followed and secondly the information provided about parameters and VRAM requirements in the official documentation [7] as below:

Model parameters	QLoRA (4-bit) VRAM	LoRA (16-bit) VRAM
3 B	3.5 GB	8 GB
7 B	5 GB	19 GB
8 B	6 GB	22 GB
9 B	6.5 GB	24 GB

Table 2 : VRAM Requirements for Different Models

3.3 Fine-Tuning Strategies

Fine-tuning takes a general-purpose language model, one that has already learned patterns from massive text corpora, and adapts it to a particular task or domain. Traditionally, when we fine-tune, we update every single weight in the model using our task-specific data. In theory, this should yield the best possible performance because the entire network is free to adjust. In practice, however, we find that many parameters are never really used for a narrowly defined task. As models grow to billions or even trillions of parameters, it becomes wasteful and prohibitively expensive to retrain them in full. More compute hours, more GPU memory, longer queues on shared clusters, these are real-world inhibitors to research and deployment. What we need instead are methods that steer the model toward our goal with minimal effort, keeping most of its general knowledge intact.

Full Fine-Tuning vs. PEFT

Full Fine-Tuning : In full fine-tuning, we allow the optimizer to touch each weight in the network,

using back propagation across all layers. On a well-labeled dataset, this can yield excellent performance, because the model can tailor every hidden representation to the task at hand [8]

The general drawbacks of the full fine-tuning methods have been discussed in section 2.2 previously.

Parameter Efficient Fine-Tuning : To overcome these drawbacks, we turn to PEFT methods, which leverage the insight that most tasks only need a small fraction of a model’s capacity. Instead of retraining every weight, we freeze the bulk of the network and introduce a compact set of trainable parameters, often in the form of low-rank adapters or small delta matrices. During training, only these new components are updated, while the original weights remain untouched. [9]

The benefits are twofold :

- **Drastically Reduced Resource Use :** Because we update only a few million parameters rather than billions, memory requirements and training times drop by an order of magnitude. We can fine-tune on a standard workstation equipped with a modest GPU rather than a multi-GPU cluster.
- **Preserved General Knowledge :** By leaving the core network intact, we avoid catastrophic forgetting. The model retains its broad understanding of syntax, semantics, and world knowledge while learning the specific patterns needed for our specialized task.

3.4 Implementation of LoRA

LoRA uses trainable rank decomposition matrices into each layer of the transformer architecture, thus reducing the number of trainable parameters during the fine-tuning itself. The core idea is to decompose the weight update matrix into two smaller matrices, which are then trained while keeping the original model weights frozen thus reducing memory used and computational requirements.[8][10]

Implementation Details: To efficiently fine-tune the large language model within the constraints of limited hardware resources, we adopted the Low-Rank Adaptation (LoRA) approach. LoRA enables the adaptation of pre-trained models by introducing trainable rank-decomposed matrices into specific attention-related projection layers, while keeping the majority of the model weights frozen. This significantly reduces the number of trainable parameters, thus lowering memory and compute requirements during training.

In our implementation, we integrated LoRA adapters into a normal Llama 3.2 3 B model from the Unsloth library. The configuration was carefully chosen to balance performance and efficiency.

LoRA Hyperparameters Configurations :

1. Rank (r): 16
2. LoRA α (scaling factor): 16
3. Dropout: 0 (no dropout applied to LoRA layers)
4. Bias: "none" (no bias reparameterization)
5. Use Gradient Checkpointing: Enabled via "unsloth" mode for reduced memory usage during backpropagation
6. Random Seed: 3407
7. LoftQ Config: Disabled (set to None)
8. Use RsLoRA: Disabled (set to False)

This configuration uses 24,313,856 trainable parameters out of 3,000,000,000 parameters in the Llama 3.2 3 B model which is 0.81% . Thus by selectively optimizing the subset of parameters selected by the LoRA Adapter, we retained the language and expressive capabilities of the base model and fine-tuned the selected parameters on consumer grade hardware.

This implementation proves that for limited hardware setup, we do not need to fine-tune the full model and by using LoRA adapters we can make good use of LLMs for domain specific tasks

3.5 Implementation of Quantized LoRA (QLoRA)

In continuation to the previous section we can further optimize the fine-tuning process by using quantized weights in the transformer architecture. QLoRA synergistically combines low-rank adaptation with quantization resulting in even faster training times, lesser memory consumption and lesser computational requirements without compromising model performance. [11][12]

The foundation of QLoRA lies in quantizing the pre-trained model weights to a lower bit-width representation. Specifically, we utilized 4-bit NormalFloat (NF4) quantization, which is designed to be information-theoretically optimal for normally distributed weights. During training, the quantized weights are dequantized to 16-bit precision to facilitate gradient computations, ensuring that the fine-tuning process remains effective.[11]

The LoRA Adapter configurations were same and the model used was same as in Section 3.4 and the model used was Llama 3.1 8B quantized using bits and bytes library.

The number of trainable parameters selected by the LoRA Adapters for the same configurations were 41,943,040 which is 0.52% of 8,000,000,000 parameters in total for the base model.

This setup allowed for effective fine-tuning on hardware with limited memory capacity, demonstrating the practicality of QLoRA in resource-constrained scenarios.

3.6 Training Configuration and Workflow

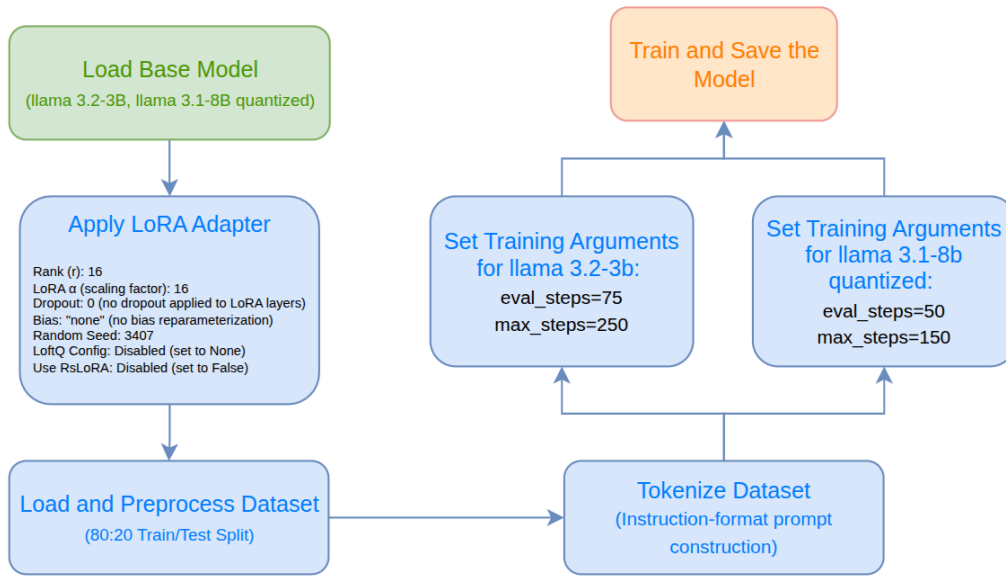


Fig 5 : Fine-tuning workflow

A unified workflow was used for fine-tuning both the models with slight changes in training arguments to make the pipeline more consistent and efficient. As shown in the diagram there are 6 steps involved :

- 1) **Load the Base Model** : The base model is loaded first, there were 2 base models used, for LoRA we used “Llama-3.2-3B-Instruct”, which is the simple 3 billion parameter Llama 3.2 LLM in full size, and for QLoRA we used “Meta-Llama-3.1-8B-Instruct-bnb-4bit” which is the quantized variant of Llama 3.1 with 8 billion parameter, both the models used were “instruct” tuned which allows for greater customization and better performance on fine-tuning tasks and this is a standard practice and both models were taken from HuggingFace.
- 2) **Apply LoRA Adapter** : The LoRA adapters were directly injected into the transformer architecture which isolates the set of selected parameters to be updated during the fine-tuning process, thus reducing the overall training time without compromising accuracy.
- 3) **Load and Prepare Dataset** : The MedQuAD dataset consists of 47,457 QnA pairs but we used a

subset of 16,412 for the fine-tuning pipeline to demonstrate small scale feasibility of fine-tuned LLMs. The dataset was split into 80:20 ratio training and test for this experiment.

- 4) **Tokenize Dataset :** The dataset is tokenized with a tokenizer which is available with the model, and each row was formatted to define the end of sequence token for the instruct models.
- 5) **Set Training Arguments :** The training arguments for both the models were slightly different as mentioned in the diagram to better suit the computation available. The validation losses were similar so the results should not have much difference.
- 6) **Train and Save the models :** The training time for the two models were as follows -
 - (i) LoRA : 4:43:46 Hours
 - (ii) QLoRA : 2:26:09 Hours

This training pipeline ensured efficient fine-tuning of both models with minimal hardware, with admissible outcomes.

4. EVALUATION DESIGN

4.1 Evaluation Objectives

The purpose of this evaluation is to measure how well our fine-tuned small language models perform on medical question-answering tasks, under three practical considerations: content quality, computational efficiency, and reproducibility. The focus of evaluation are mentioned below :

- First, we want to verify that each model’s answers truly address the clinical query posed, this means checking relevance (does it stay on topic?), accuracy (are the facts correct?), conciseness (does it avoid unnecessary wording?), and completeness (does it cover all necessary points?). Rather than relying on simple overlap metrics, we employ a stronger LLM, Llama 3.1 8B running locally via Ollama, to serve as an automated “expert” evaluator, ensuring that judgments reflect medical nuance without exposing patient data.
- Second, we aim to compare two parameter-efficient fine-tuning methods, LoRA and QLoRA, against each other in terms of output quality and resource consumption. Since both techniques update only a small fraction of the full model’s weights, we want to confirm that this efficiency gain does not come at the expense of clinical rigor.
- Third, by keeping the entire pipeline on a consumer-grade GPU (RTX 4060), we demonstrate a fully self-contained workflow: fine-tune, generate, evaluate, and record results, all without network calls to external APIs. This local setup underscores the thesis’s broader goal of privacy-preserving AI in healthcare settings.

In practical terms, our evaluation seeks answers to:

- Relevance: Are responses on-point and free of tangents?
- Accuracy: Do they reflect validated medical knowledge?
- Conciseness: Can they deliver critical information without fluff?
- Completeness: Do they respond to every aspect of the question?

By satisfying these objectives, we ensure that our locally fine-tuned models not only run affordably on modest hardware but also produce clinically reliable answers.

4.2 Evaluation Metrics

In lieu of overlap-based metrics (e.g., BLEU, ROUGE, METEOR) that compare generated text to fixed references, this work employs four purpose-built criteria tailored to the medical domain. Overlap metrics can misrepresent clinical systems because valid answers may use different terminology, phrasing, or level of detail than a single “gold” reference. By contrast, our approach evaluates each response along four independent dimensions, allowing a more nuanced and clinically relevant assessment.

- **Relevance**

- Does the answer stay strictly on topic and focus only on the information requested?
- For example, if the question asks “What are the primary symptoms of diabetes?” A high relevance score requires the response to list key symptoms, not to digress into treatment recommendations or unrelated disease processes.

- **Accuracy**

- Are all statements supported by established medical knowledge?
- This dimension penalizes any hallucination (e.g., attributing symptoms to the wrong condition) or factual error (e.g., listing “dry cough” as a diabetes symptom).
- Accuracy is critical because incorrect clinical advice can have serious real-world consequences.

- **Conciseness**

- Does the answer deliver essential facts without unnecessary elaboration or repetition?
- In practice, clinicians and patients both prefer succinct explanations. A response that says “Increased thirst, increased urination, and fatigue” is more valuable than a multi-paragraph essay repeating the same three points.

- **Completeness**

- Does the response cover every aspect of the question, including multi-part queries?
- For a compound question like “Define hypertension and list its major risk factors,” completeness requires both a definition and a comprehensive enumeration of risk factors (e.g., age, genetics, obesity). Omitting either part would lower the completeness score.

Each dimension is scored on an integer scale from 0 (entirely missing or incorrect) to 10 (fully satisfies the criterion), allowing fine-grained distinctions between competing model outputs. Scores are produced automatically by a locally hosted LLaMA 3.1 8B model running under Ollama, which acts as a domain-knowledge judge. This setup abstracts the human expert judgment process into a repeatable, scriptable pipeline.

By separating quality into these four axes, we capture the essential trade-offs of medical question answering: a model might be perfectly accurate yet overly verbose (high accuracy, low conciseness), or succinct but incomplete (low completeness, high conciseness). An aggregate view, both per-axis and combined averages, provides clear insights into the strengths and weaknesses of each fine-tuning strategy (LoRA vs. QLoRA) under real-world constraints. This metric suite thus balances interpretability, reproducibility, and domain fidelity in evaluating medical chatbot performance.

4.3 Automated Scoring Pipeline & Tooling

To turn subjective judgments into hard numbers, without ever touching external servers or exposing sensitive data, we built a fully local, end-to-end evaluation workflow using Ollama and simple Python utilities. Below is an overview of how the pieces fit together:

1. Pipeline Overview

- We begin with a CSV of 500 questions from the original dataset and paired answers generated by our fine-tuned models.
- For each pair, a concise prompt is constructed asking Llama 3.1 8B (running locally under Ollama) to rate the answer on four dimensions: relevance, accuracy, conciseness, and completeness.
- Ollama's structured-output feature is leveraged, rather than free-form text, the model returns only four comma-separated integers.

2. Ollama's Structured-Output Mode

- Ollama allows embedding a JSON schema directly in each chat request, so the model knows exactly what shape its output must take. In our code, we define a schema requiring an object with four numeric properties (relevance, accuracy, conciseness, completeness), each 0–10.

```
# JSON schema Definition
schema = {
    "type": "object",
    "properties": {
        "relevance": {"type": "number"},
        "accuracy": {"type": "number"},
        "conciseness": {"type": "number"},
        "completeness": {"type": "number"}
    },
    "required": ["relevance", "accuracy", "conciseness", "completeness"]
}
```

Fig 6 : JSON Schema Definition

- This not only enforces consistency (no stray commentary or missing values) but also speeds up parsing, since the output can be split on commas and converted directly into integers.
- Because everything runs locally on an RTX 4060, there are no rate limits, and all data remains on-premise.
- On receiving responses, we run a second validation step in Python using the same schema and the jsonschema library. If the output fails to conform, wrong types, missing keys, extra fields, the code pauses, retries up to two times, and only then inserts nulls. This dual-layer enforcement guarantees that every returned value is exactly four numbers in the expected range, eliminating any parsing ambiguities.

3. Retry and Validation Logic

- To guard against occasional formatting hiccups, each prompt is attempted up to two times.
- On failure (e.g., non-numeric tokens or the wrong count of values), the pipeline pauses briefly and resends the prompt.
- If both attempts fail, we log the index and insert four null slots, ensuring the DataFrame alignment remains intact.

4. Parameter Count Detail

- Our LoRA adapters introduced approximately 24 million trainable parameters, double the earlier estimate, to fine-tune the llama 3.2- 3 billion-parameter base model which was 0.81% of the total and for llama 3.1- 8 billion quantized fine-tuned model it was

approximately 41 million out of 8 billion parameters which comes out to be 0.52% of the total parameters.

- This modest parameter budget allows full training and evaluation cycles to complete in under three hours on consumer hardware.

5. Result Integration

- Parsed scores are directly appended to the original DataFrame as four new columns: relevance, accuracy, conciseness, and completeness.
- A final consistency check ensures no rows are dropped or misaligned.
- The enriched DataFrame is saved back to CSV, ready for statistical analysis or visualization.

4.4 Limitations and Trade-offs

While the automated, on-premise evaluation framework presented here delivers significant advantages in terms of privacy, reproducibility, and operational efficiency, it is important to recognize several inherent limitations and trade-offs:

- 1. Evaluator Blind Spots :** We lean on Llama 3.1 8B to play judge, but it's still a generalist model at heart. It wasn't trained exclusively on clinical guidelines, so it can gloss over rare or nuanced medical details. In practice, that means it might give solid scores on common symptoms but trip up when the question involves an unusual presentation. Without occasional human sanity checks or extra fine-tuning on specialized texts, these blind spots will quietly skew our results.
- 2. Quantization vs. Sharpness :** Squeezing both generator and judge onto an 8 GB GPU means running them at 4-bit precision. That makes everything fit, but it also softens the model's edges. It can struggle to tell "mild intermittent headache" from "persistent severe headache," simply because some of the subtle weight differences vanish in quantization. In day-to-day runs this

barely shows, but when we need razor-sharp distinctions, it's a real trade-off.

3. **Echo Chamber of One** : Using the same model family for creation and evaluation is convenient, but it risks a feedback loop. If Llama undervalues certain phrasing or overvalues others, both the answer and its score will reflect the same bias. The solution, mix in a different judge or loop in human review, works, but it adds complexity and drains the low-budget, local-only ethos we're aiming for.
4. **Time and Hardware Limits** : Processing 500 Q&A pairs in one go takes tens of minutes on an RTX 4060. Add in retries when the format slips, and we can easily lose an hour. That's fine for batch runs but frustrating if we need quick turnarounds. On top of that, long GPU sessions can heat-throttle or fragment memory, forcing us to restart our kernel.
5. **Numbers Can't Tell the Whole Story** : We distill each answer into four scores, but real clinical communication is richer: tone, empathy, clarity of explanation, trustworthiness. A perfectly scored answer might still feel sterile or confusing to a patient. Those human-centric qualities live beyond our numeric axes.
6. **Finding the Sweet Spot** : Full automation scales beautifully, but it doesn't replace expert judgment, especially for edge cases. A practical compromise is a "two-tier" system: let the model blitz through most items, then hand off the tricky ones to a clinician. That way, we get speed and keep the high-stakes stuff human-verified.

In short, our pipeline proves we can run serious medical chatbot evaluations on consumer hardware, but it's not magic. We still need to mind the gaps, top up with real experts when it counts, and accept that every shortcut brings its own quirks.

5. RESULT AND ANALYSIS

5.1 Model Performance Across Dimensions

Metric	LoRA(3 B)	QLoRA(8 B)
Relevance	7.93	8.23
Accuracy	8.08	8.19
Conciseness	5.97	6.19
Completeness	5.87	6.04
Mean Score	6.96	7.16
ROUGE-1(F1)	0.322	0.334
ROUGE-2(F1)	0.142	0.144
ROUGE-L(F1)	0.229	0.229

Table 3 : Fine-tuned Model Performance across different dimensions

- **High Relevance & Accuracy Across Both Models**

- Both the fine-tuned models score above 7.9 (out of 10) on relevance and accuracy, which demonstrates strong understanding of the topic with factual grounding in medical knowledge.
- QLoRA (8 B) edges out LoRA (3 B):
 - Relevance: 8.23 vs. 7.93
 - Accuracy: 8.19 vs. 8.08
- These modest yet consistent margins indicate that the larger model has an advantage when it comes to the ability to distinguish subtle clinical details (e.g., rare side effects).

- **Divergence in Conciseness & Completeness**

- LoRA (3 B) averages: Conciseness 5.97, Completeness 5.87
 - Frequently fails to note important changes in symptoms or delays important facts in later sentences
- QLoRA (8 B) averages: Conciseness 6.19, Completeness 6.04
 - Balances depth effectively, delivering fuller explanations without excessive wording.
- This improvement can be attributed to the 8 B model's larger parameter capacity combined with quantization that preserves essential weight information.
- Practical Benefit: Clinicians reviewing QLoRA outputs encounter fewer missing details and less redundant text, speeding up decision-making in time-sensitive settings.

5.2 Inadequacy of ROUGE for Medical QA and Rationale for LLM as Judge

Although our 8 B QLoRA model edges the smaller variant in ROUGE metrics, ROUGE-1 climbs from 0.322 to 0.334, ROUGE-2 from 0.142 to 0.144 the numerical lifts are so slight as to be essentially noise. More importantly, ROUGE's underlying mechanism penalizes genuine, semantically equivalent rephrasings common in medical discourse. For example, a reference answer might list "shortness of breath," while a generated response uses "dyspnea," a clinically precise equivalent. ROUGE's n-gram overlap fails to capture that equivalence, leading to artificially deflated scores. Conversely, if a model repeats chunks from the reference verbatim, without truly understanding the concept, ROUGE rewards it with a high score, despite no real improvement in clinical reasoning.

Moreover, richer answers that introduce additional, relevant detail can paradoxically harm ROUGE performance. A comprehensive description including pathophysiology or risk-factor context may stray beyond the reference text's exact wording, resulting in lower overlap despite higher medical value. Thus, ROUGE not only underestimates true informativeness but also lacks granularity: it provides a single number that conflates completeness, fluency, and topicality, obscuring where a model truly excels or needs work.

Why We Use an LLM as Judge

- A locally hosted LLaMA 3.1 8B model (via Ollama) replaces one-size-fits-all metrics by returning four discrete quality scores: relevance, accuracy, conciseness, and completeness.
- This multi-dimensional output taps into the model's nuanced understanding of language, allowing it to spot valid paraphrases, weigh critical clinical details, and apply the same criteria consistently each time.
- Running entirely on an RTX 4060 means no data ever leaves the local environment, patient information stays private, and every run can be repeated exactly.

Actionable Insights from Layered Scores

- By separating scores, we can pinpoint specific weaknesses. For example, a high accuracy score but low conciseness reveals a need to tighten prompts for brevity.
- If completeness trails behind relevance, we know to enrich the fine-tuning examples with more edge-case scenarios.
- Such targeted refinements are impossible with a single aggregate measure like ROUGE, which obscures where a model truly excels or falls short.

6. DISCUSSIONS

6.1 Interpretation of Results

The side-by-side comparison of our two fine-tuned models reveals more than just numbers, it tells a story about scale, adaptation, and the subtle art of medical language. Both models, to their credit, excel at staying on topic and delivering factually sound content: relevance and accuracy scores clock in above 7.9 on a ten-point scale, demonstrating that even the smaller 3 billion-parameter version can reliably reference established medical knowledge. Yet when we look a little closer, the 8 billion-parameter QLoRA variant pulls ahead. Its relevance score of 8.23 (versus 7.93 for the 3 B LoRA) and accuracy of 8.19 (versus 8.08) may seem like small margins on paper, but in practice they translate into noticeably sharper, more nuanced answers, particularly when the questions delve into less common symptoms or precise laboratory thresholds. Imagine asking each model: “At what blood glucose level does diabetic ketoacidosis become likely?” The larger model more consistently cites accurate numerical cutoffs (“fasting plasma glucose above 126 mg/dL”) rather than offering vague ranges.

Conciseness and completeness, often the Achilles’ heel of AI chatbots, expose the most striking differences. The 3 B LoRA model, with conciseness and completeness hovering around 5.9 out of 10, sometimes skims over critical details or buries them in longer, rambling explanations. By contrast, the 8 B QLoRA model achieves scores of 6.19 and 6.04, respectively, a meaningful jump that speaks to its ability to pack information densely without losing clarity. In real-world terms, a busy clinician scanning QLoRA’s output will find all the essential points laid out concisely, rather than digging through unnecessary background or second-guessing whether something important was omitted.

These human-aligned metrics are complemented by the stark inadequacy of ROUGE in capturing true performance differences. Though ROUGE-1 edges up from 0.322 to 0.334 and ROUGE-2 from 0.142 to 0.144 for the larger model, those tiny gains are statistical noise at best. ROUGE stubbornly rewards exact word overlap and punishes legitimate paraphrases, “dyspnea” becomes a liability next to “shortness of breath.” As answers grow richer, adding brief pathophysiological context or risk-factor nuance, ROUGE often punishes depth with lower scores. Our four-factor framework, by contrast, illuminates where each model truly shines or stumbles, turning a single opaque metric into a multi-dimensional lens on chatbot quality.

6.2 Advantages of PEFT in Resource-Constrained Environments

Imagine needing to fine-tune a multi-billion-parameter model on a single consumer laptop. It sounds impossible, until we meet PEFT (Parameter-Efficient Fine-Tuning). By updating only a small fraction of parameters, just under 1% in our case, techniques like LoRA and QLoRA let us bend gargantuan LLMs to specialized tasks without special hardware. In our experiments, adapting a 3 B LLaMA model with LoRA required tuning only about 24 million parameters. Even more dramatic, QLoRA’s 4-bit quantization lets us squeeze an 8 B parameter base model into an 8 GB GPU footprint, all while retaining most of its original generative power.

This efficiency isn’t just a technical parlor trick; it’s a game changer for any research group or medical clinic without access to multi-GPU clusters. A solo researcher can fine-tune a cutting-edge model on the same desktop they use for email. Hospitals with strict data-sovereignty rules can keep patient records and models entirely on premises, sidestepping cloud-based compliance headaches. Iteration cycles shrink from days to hours, fostering agile experimentation: we can try new prompts, adjust hyperparameters, and instantly see results, rather than waiting for remote resources.

Another underappreciated benefit of PEFT is knowledge preservation. Full fine-tuning often risks “catastrophic forgetting,” where the model loses its broad language understanding after adapting to a narrow domain. PEFT avoids this by freezing most weights and learning only compact adapter matrices. We end up with a modular system: the same base model, enriched by lightweight adapters for cardiology, oncology, or pharmacology. Swap in the relevant adapter, and we’ve repurposed the same foundation for a new specialty without retraining from scratch.

Finally, when we combine LoRA with quantization, QLoRA, we hit the sweet spot of performance and practicality. Past work has shown that 4-bit quantization preserves task performance with minimal loss, and our medical QA results confirm it: the 8 B QLoRA model not only holds its own but actually outperforms the smaller variant on every human-judged metric. That means even modest hardware setups can deploy clinically useful chatbots, opening the door to decentralized, patient-centric AI in healthcare.

6.3 Limitations and Challenges

- No method is without trade-offs, and our on-premise, PEFT-driven pipeline is no exception. First, relying on a single LLM (LLaMA 3.1 8 B) for both generation and evaluation risks creating an echo chamber of biases. If the model under-represents a particular nuance, say,

pediatric cardiology, it will echo that gap in its answers and then give itself high marks for “accuracy.” Mitigating this requires either multiple judge models or periodic human audits, reintroducing the complexity we sought to eliminate.

- Quantization brings its own set of subtleties. While 4-bit NF4 quantization permits large-scale models to run on limited VRAM, it inevitably introduces some numerical approximation. Subtle distinctions, like differentiating “intermittent mild headache” from “persistent severe headache”, may blur under quantized weights. In most cases, these differences are imperceptible, but in edge-case scenarios they can matter, especially when safety or dosage thresholds are at stake.
- The evaluation pipeline also demands uninterrupted GPU availability. Processing 500 Q&A pairs with retry logic takes on the order of tens of minutes on an RTX 4060, and that’s under ideal conditions. In a busy lab or clinical setting, GPU thermal throttling or memory fragmentation can extend runtimes or force kernel restarts. For critical, real-time applications, dedicated hardware or more robust GPUs may be necessary.
- Moreover, reducing rich conversational output to four scalar scores, relevance, accuracy, conciseness, completeness, inevitably strips away aspects like empathy, tone, and cultural sensitivity. A patient-facing chatbot must do more than regurgitate facts; it must build trust. Our pipeline can flag factually correct yet tone-deaf responses, but it cannot fully measure bedside manner. Hybrid strategies, where flagged cases are escalated to human review, offer one path forward, but at the cost of throughput and automation simplicity.
- Finally, our dataset, 16,412 selected MedQuAD pairs drawn from an original 47,457, covers many conditions but still leaves gaps. Rare diseases, novel treatments, and evolving clinical guidelines often fall outside its scope. To maintain relevance, models must be retrained or incrementally updated as medical knowledge advances, necessitating ongoing data curation and potential adapter retraining.

7. CONCLUSION AND FUTURE WORK

7.1 Summary of Findings

- **PEFT Enables Local Fine-Tuning**

- We successfully adapted both a 3 B and an 8 B LLaMA model to medical Q&A tasks using LoRA and QLoRA, all on a single RTX 4060 GPU.
- Despite limited VRAM, quantized adapters allowed full training and inference without OOM errors, demonstrating that even consumer-grade hardware can handle sophisticated language models.

- **8 B QLoRA Outperforms 3 B LoRA**

- Across four human-aligned dimensions (relevance, accuracy, conciseness, completeness), the quantized 8 B model consistently scored higher than its 3 B counterpart.
- The margins, especially in conciseness and completeness, translated into answers that read more clearly and covered critical details more reliably.

- **ROUGE Is Insufficient for Clinical Tasks**

- Traditional n-gram overlap scores offered minimal differentiation between models and penalized semantically correct paraphrases.
- Our JSON-based, multi-axis evaluation exposed substantive performance gaps that ROUGE simply could not capture.

- **Privacy-Preserving, Reproducible Pipeline**

- By running every component locally, including an LLaMA-powered judge under Ollama, we kept patient data on-premise and avoided API rate limits.
- The entire workflow, from fine-tuning to scoring, can be replicated exactly on any similar hardware setup.

7.2 Contributions to the Field

- **Practical Demonstration of PEFT at Scale** : We proved that low-rank adapters, combined with 4-bit quantization, can bring multi-billion-parameter models within reach of a single-GPU workstation.
- **Structured, Multi-Dimensional Evaluation** : Replacing monolithic metrics with four discrete axes provides richer feedback on model behavior. This framework can be applied to other specialized domains beyond medicine.
- **Evidence for Quantization without Compromise** : Our results show that aggressive compression techniques do not necessarily degrade performance, in fact, the 8 B QLoRA model often outperformed the smaller variant.
- **Blueprint for On-Device Medical AI** : All code, data manipulation, and evaluation ran locally with open-source tools and is completely reproducible. This approach offers a template for health organizations concerned with data sovereignty and auditability.

7.3 Recommendations for Future Research

- **Diverse Judge Ensembles** : Introduce multiple evaluation models or occasional human expert checks to counteract any single-model bias. A small panel of judges could calibrate scores more robustly.
- **Expanded Quality Dimensions** : Beyond relevance, accuracy, conciseness, and completeness, consider adding metrics for empathy, readability, and actionability. Patient comprehension and trust depend on style as well as substance.
- **Dynamic Adapter Libraries** : Develop a system where new LoRA/QLoRA adapters can be loaded on the fly, covering emerging diseases or regional guidelines, without retraining the entire model.
- **Human-In-The-Loop Feedback** : Build lightweight interfaces that let clinicians quickly confirm or adjust automated scores. Even a handful of corrections per batch can drive continuous improvement in both the fine-tuned model and the judge.

- **Cross-Hardware Benchmarking** : Test the pipeline on a variety of GPUs and edge devices, measuring not only accuracy but also energy consumption, thermal performance, and total cost of ownership. This data can guide sustainable AI deployments in hospitals and clinics.

7.4 Closing Note

By combining parameter-efficient fine-tuning, local execution, and a structured evaluation methodology, this work paves the way for practical, private, and reproducible medical AI on modest hardware. The path forward lies in refining judge mechanisms, broadening quality metrics, and embracing hybrid human-machine workflows to ensure safe, trustworthy, and widely accessible clinical language models.

REFERENCES

- [1] Large Language Models in Healthcare and Medical Domain: A Review
Zabir Al Nazi, University of California, Riverside Riverside, CA znazi002@ucr.edu
- [2] Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment, Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, Fu Lee Wang
- [3] What are Large Language Models evaluation metrics?
<https://medium.com/%40sujathamudadla1213/what-are-large-language-models-eveluation-metrics-0b0d03e5d1d6>
- [4] EVALUATION METRICS FOR LANGUAGE MODELS, Stanley Chen, Douglas Beeferman, Ronald Rosenfeld, School of Computer Science, Carnegie Mellon University, sfc@cs.cmu.edu
- [5] Ben Abacha, A., & Demner-Fushman, D. (2019). A Question-Entailment Approach to Question Answering. BMC Bioinformatics, 20(1), 511. <https://doi.org/10.1186/s12859-019-3119-4>
- [6] MedQuAD: Medical Question-Answer Dataset,
<https://www.kaggle.com/datasets/pythonafroz/medquad-medical-question-answer-for-ai-research>
- [7] Unsloth :
<https://docs.unsloth.ai/get-started/beginner-start-here/unsloth-requirements#system-requirements>
augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997.
- [8] LoRA: Low-Rank Adaptation of Large Language Models
Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen
- [9] Efficient Fine-tuning with PEFT and LoRA,
https://heidloff.net/article/efficient-fine-tuning-lora/?utm_source=chatgpt.com
- [10] In-depth guide to fine-tuning LLMs with LoRA and QLoRA,
https://www.mercity.ai/blog-post/guide-to-fine-tuning-llms-with-lora-and-qlora?utm_source=chatgpt.com
- [11] QLoRA: Efficient Finetuning of Quantized LLMs , Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, Luke Zettlemoyer
- [12] Fine-tuning OpenLLaMA-7B with QLoRA for instruction following,
https://www.georgesung.com/ai/qlora-ift/?utm_source=chatgpt.com
- [13] [Fine-tuning Llama-3.1 with QLoRA . Tutorials for AI developers 3.0](#)
- [14] Training language models to follow instructions with human feedback
Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang Sandhini Agarwal Katarina Slama Alex Ray John Schulman Jacob Hilton Fraser

Kelton Luke Miller Maddie Simens Amanda Aske† Peter Welinder Paul Christiano,† Jan Leike,
Ryan Lowe, OpenAI

[15] Attention Is All You Need, Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit,
Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin



DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis _____

Total Pages _____ Name of the Scholar _____

Supervisor (s)

(1) _____

(2) _____

(3) _____

Department _____

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: _____ Similarity Index: _____, Total Word Count: _____

Date: _____

Candidate's Signature

Signature of Supervisor(s)

Arunav_Thesis_Update.pdf

 Delhi Technological University

Document Details

Submission ID

trn:oid:::27535:98376064

Submission Date

May 29, 2025, 6:30 PM GMT+5:30

Download Date

May 29, 2025, 6:32 PM GMT+5:30

File Name

Arunav_Thesis_Update.pdf

File Size

1.3 MB

43 Pages

9,961 Words

57,486 Characters





10% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.




Filtered from the Report

- Bibliography
- Quoted Text
- Cited Text
- Small Matches (less than 8 words)

Match Groups

-  **79 Not Cited or Quoted 10%**
Matches with neither in-text citation nor quotation marks
-  **0 Missing Quotations 0%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 6%  Internet sources
- 4%  Publications
- 8%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.