Fine-Tuning for Medical Consultation Response Generation: A Comparative study of Baseline, LoRA, and Full Fine-Tuning on Physician-Patient Dialogues

Ekom Etuk
Data Science and Computational Intelligence
Coventry University
Coventry, England
etuke2@coventry.ac.uk

This study investigates parameter-efficient fine-tuning of a 4-bit quantized Mistral-7B-Instruct model for medical dialogue generation using the MedDialog dataset. Low-Rank Adaptation (LoRA) was applied with rank 16, reducing trainable parameters from 7 billion to 40 million (0.57% of the original model). Training and validation losses decreased from 1.858187 and 1.839446 to 1.697663 and 1.730888 respectively over 400 steps, plateauing at ~ 1.73 which informed early stopping at 450 steps. Evaluation revealed significant improvements in lexical metrics, with ROUGE-2 increasing by 105.3% and ROUGE-L by 53.9%, while semantic metrics showed slight degradation.

Keywords— Large Language Model, LoRA, Generative AI, Parameter Efficient Fine-Tuning.

I. INTRODUCTION

The global Artificial Intelligence (AI) in healthcare market capitalization is projected to grow from $56.01 billion in 2026 to over $1 trillion by 2034, driven by demands for efficiency amid physician shortages and rising patient volumes. Large language models (LLMs) are revolutionizing this sector by automating clinical tasks like response generation, medical reports summarization, language translation, with studies showing generative AI drafts reduce provider turnaround time while matching human empathy and accuracy [1]

Healthcare faces acute challenges, ranging from global physician shortages which is expected to exceed 18 million by 2030, to rapid response time while patient-physician messages surge 20-30% annually, overwhelming systems. AI addresses this by scaling consultations and helpfulness in dialogues, enabling 24/7 triage and personalized advice without replacing clinicians. This paper compares baseline, full fine-tuning (FFT), and LoRA on physician-patient dialogues. We use Mistral-7B model on MedDialog dataset, providing reproducible code and ethical analysis of biases, thereby paving parameter-efficient paths for real-world deployment.

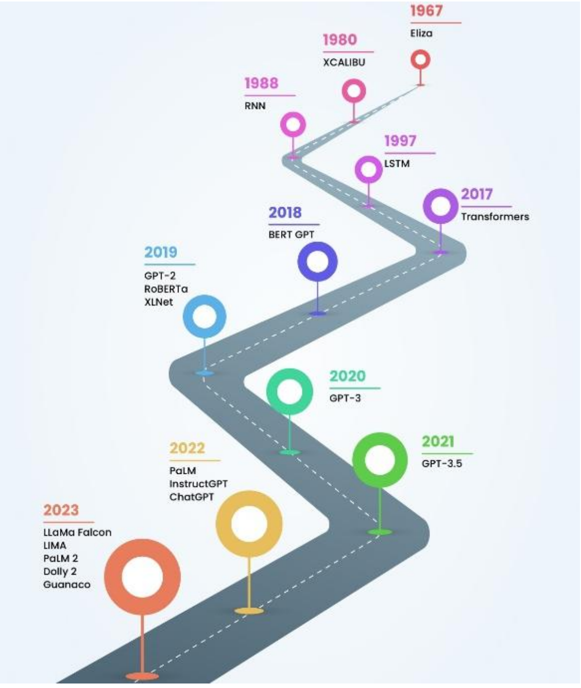
II. LITERATURE

A. A Brief History of LLMs

The conceptual foundations of conversational AI were laid decades before their modern realization, beginning with symbolic and rule-based systems like Eliza in the 1960s and XCALIBU in the 1980s. A fundamental paradigm shift occurred with the move towards neural architectures, marked by the introduction of Recurrent Neural Networks (RNNs) in 1988 and Long Short-Term Memory networks (LSTMs) in 1997, which enabled models to process sequential data more effectively. A watershed moment arrived in 2017 with the publication of the seminal paper *Attention Is All You Need* by Google researchers, which introduced the Transformer architecture [2]. The Transformer unlocked the potential of Pre-trained Language Models (PLMs). In 2018, OpenAI released Generative Pre-trained Transformer (GPT-1) [3], later that year, Google introduced Bidirectional Encoder Representations from Transformers (BERT) [4], BERT's deep bidirectional understanding led to state-of-the-art results on a wide array of NLP tasks, thereby setting a new standard for language representation, this was followed by a series of advancements including GPT-2, RoBERTa, and XLNet in 2019.

The subsequent years witnessed rapid scaling and refinement of these models. In 2020, GPT-3 demonstrated the power of massive scale with its 175 billion parameters, setting new benchmarks in few-shot learning. This was followed by GPT-3.5 in 2021 and a surge of innovations in 2022, including PaLM, InstructGPT, and ChatGPT, which highlighted the importance of instruction tuning and reinforcement learning from human feedback. The year 2023 saw an explosion of diverse models such as LLaMA, Falcon, LIMA, PaLM 2, Dolly 2, and Guanaco, reflecting a trend toward more efficient, accessible, and specialized architectures.

Figure 1: Evolution of LLMs



Source : https://medium.com/@mail2alamuram/large-language-model-in-simple-terms-06ed8752ad70

B. Large Language Models in Healthcare

The continuous development and integration of Large Language Models (LLMs) into healthcare has shifted the landscape of medical report and dialogue summarization, medical consultation, and day-to-day communication. LLMs like Bio_ClinicalBERT can handle medical texts and report summarization efficiently on a large scale within a very short period of time. Fine-tuned models like Med-PaLM 2 and BioMistral, are no longer just information retrieval tools but active agents capable of generating long-form answers to bedside consultation questions that physicians often prefer over generalist responses due to their depth and alignment with clinical needs [5].

However, the integration and deployment of LLMs into clinical practice presents significant hurdles, ranging from regulatory and legal governance to technical vulnerabilities such as hallucination and data poisoning [6].

C. The Role of Parameter-Efficient Fine-Tuning (PEFT) in Medical LLMs

Adapting massive foundational models to the medical domain presents a computational bottleneck. As model sizes grow, full fine-tuning becomes increasingly expensive to store, train, and deploy across multiple tasks or domains. Fine-tuning (FFT) of billion-parameter models is resource-intensive. Parameter-Efficient Fine-Tuning (PEFT) has emerged as a vital solution, enabling the adaptation of LLMs by updating only a small fraction of parameters. Computationally realistic, about 120GB of GPU VRAM would be required to full-finetune a 7-Billion parameter model in a 16-bit precision. PEFT methods like Low-Rank Adaptation (LoRA) and Quantized Low-Rank Adaptation (QLoRA) address LLM fine-tuning's compute barriers. This is done by introducing trainable low-rank matrices while freezing pretrained weights, reduced trainable parameters from 7 billion to roughly 104 million i.e a ~98% reduction while achieving performance comparable to FFT [7].

III. PROBLEM STATEMENT

Patient–provider messaging and telemedicine have increased the demand for timely, accurate, and empathetic clinical responses, clinician time is however limited and response quality can vary across providers and settings. Medical dialogue generation systems aim to support this workflow by producing medically-acceptable responses conditioned on a patient's symptoms and history. However, deploying high-performing large language models (LLMs) for this task remains challenging because (i) generic pretrained models often lack domain-specific clinical phrasing and safety-aware reasoning, (ii) hallucinated or overly confident outputs can be harmful in medical contexts [8], and (iii) full fine-tuning of modern LLMs is computationally expensive, making iterative experimentation and domain adaptation difficult under typical academic GPU constraints. In this paper, we attempt to adapt an open-source LLM to generate medically appropriate consultation responses from physician–patient dialogues while balancing response quality with computational feasibility. This motivates a comparative evaluation of (1) a baseline LLM, (2) full fine-tuning where feasible, and (3) a parameter-efficient fine-tuning method that reduces training cost while aiming to preserve or improve response quality.

IV. MODEL AND DATASET

A. Model Card

We adopt Mistral-7B-Instruct-v0.2 model as our foundation large language model for all experiments. Mistral-7B-Instruct-v0.2 is an instruction-tuned large language model (LLM) released by Mistral AI in 2023. It is a 7.3 billion parameter model built on the Transformer architecture and is an improved version of the original Mistral-7B-Instruct-v0.1. The instruct-tuned nature of this model makes it particularly suitable for medical consultation response generation, as it inherently understands dialogue structure and instruction following. The model is further described with the following properties.

Table 1: Model Properties

Team	Mistral AI
Model size	7B parameters
Released	2023
Precision	BF16
Architecture	Transformer
Context window	32,000
Supported language(s)	English
Average monthly downloads	About 2m
Url	https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2

B. Dataset Card

We employ the MedDialog dataset from OpenMed, which contains structured physician-patient dialogues. MedDialog is a curated medical dialogue dataset for physician–patient conversations, it is arguable the largest dataset of physician-patient conversation with more than 250k English dialogues across diverse specialties [9].

Table 2: Dataset Properties

Team	OpenMed
Dataset size	285 MB
Released	2020
No. of rows	251,731
No. of tokens	~ 60,000,000
Url	https://huggingface.co/datasets/OpenMed/MedDialog

V. METHODOLOGY

In this section, we describe in detail, step by step, the end-to-end experimental pipeline used to compare baseline inference, full fine-tuning, and parameter-efficient fine-tuning using LoRa for medical consultation on physician–patient dialogues.

A. Baseline Model

We use the open-source Mistral-7B-Instruct-v0.2 as the backbone and baseline LLM, evaluated directly without weight updates and loaded its paired tokenizer using the HuggingFace *AutoTokenizer* method. To make training feasible on the limited GPU memory, the base model is loaded through the AutoModelForCausalLM.from_pretrained(...) with a quantization configuration and automatic device placement. Finally, we report the backbone's parameter scale and establish a reproducibility checkpoint by computing (i) the total parameter counts and (ii) the trainable parameter counts.

B. Dataset preprocessing and Tokenization.

We acquire the physician–patient corpus via the HuggingFace (HF) using the HF dataset object and then construct a controlled training subset for experimentation. We create a reproducible train–validation partition using, yielding 85% training samples and 15% validation samples. This split is fixed by the random seed to ensure reproducibility.

All text sequences are tokenized using the Mistral-7B-Instruct-v0.2 tokenizer. Truncation was enabled at a maximum sequence length of 512 tokens to fit GPU memory constraints and ensure batch stability during training. Sequences shorter than 512 tokens are padded to max_length using the EOS token, with padding appended to the right. Tokenization is applied in batch mode to both training and validation splits, removing original text columns and retaining only tokenized input_ids, attention_mask, and related tensor fields required by the model. This tokenization pipeline ensures uniform input shapes for efficient training, eliminates variable-length sequence overhead, and maintains compatibility with the causal masking convention used in decoder-only Transformers.

C. LoRA configuration

To enable parameter-efficient adaptation of Mistral-7B-Instruct-v0.2 on the medical dialogue task, we apply Low-Rank Adaptation (LoRA) using the Hugging Face PEFT library. LoRA freezes the pretrained backbone weights and injects trainable low-rank decomposition matrices into selected Transformer modules, substantially reducing the number of parameters that require gradient updates and storage while preserving model expressiveness. Each adapter matrix is decomposed into a product of two low-rank matrices with inner dimension, balancing adaptation capacity and parameter efficiency. The learned low-rank updates are scaled before being added to the frozen weights; this is done to control the magnitude of adaptation relative to the base model. LoRA adapters are attached to all attention projection layers and MLP gating/projection layers ensuring that both self-attention and feedforward sublayers benefit from task-specific adaptation. A dropout rate of 10% is applied to the LoRA layer outputs during training to regularize the low-rank updates and mitigate overfitting on the limited training subset. Bias terms in the target modules remain frozen, simplifying the adapter configuration and keeping trainable parameters strictly within the low-rank matrices. This configuration reduces trainable parameters to about 1% of the full model size, enabling training on GPUs with limited VRAM while maintaining competitive performance relative to full fine-tuning.

Table 3: LoRa Hyperparameter Table

LoRa parameters	Value
Rank (r)	16
Scaling Factor	32
Target Modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj
Dropout	0.1
Bias	"none"

VI. EXPERIMENTAL SETUP

In this section, we report the computational environment, training configuration, and evaluation protocol used to (i) evaluate the pretrained baseline model, (ii) attempt full fine-tuning where feasible, and (iii) train a Parameter Efficient Fine-tuning using LoRA.

A. Hardware and Software Environments

All experiments described in this paper were conducted using the Python programming language with Google Colab Pro A100 GPU with libraries such as pandas numpy, datasets, transformers etc. Experiments were executed on a single GPU runtime with automatic device placement, and the base model is loaded with a bitsandbytes quantization configuration to reduce memory footprint.

B. Model Input and Quantization

The model is trained and evaluated in an autoregressive (causal LM) setting, where each training example is converted into a single text sequence that contains (i) the patient's condition description and dialogue history and (ii) the target physician response. A consistent prompt template is used to standardize inputs across the models, e.g.:

Patient: <patient_condition_or_question>

Doctor: <doctor_response>

To efficiently handle the computational demands of fine-tuning a 7B-parameter model on limited hardware, we employ 4-bit quantization and structured input formatting. The base model, Mistral-7B-Instruct-v0.2, is loaded using the bitsandbytes library with a 4-bit quantization configuration. This technique reduces the model's memory footprint by approximately 4x compared to full 16-bit precision, allowing the weights to fit within consumer-grade GPU VRAM while performing computations in 16-bit float for stability.

C. Dataset Tokenization

We tokenized the MedDialog training text using the Mistral-7B-Instruct-v0.2 tokenizer to ensure full compatibility between the dataset encoding and the base model's vocabulary. Each preprocessed example, containing the patient's query and physician response concatenated into a single text field, is tokenized with strict length constraints. This fixed sequence length enforces uniform batch shapes across all training and validation examples, stabilizing GPU memory consumption and enabling fair runtime. Long dialogues exceeding 512 tokens are truncated to retain the most recent context, which typically contains the information most relevant for generating the next physician's response. Tokenization is performed via batched mapping over both the training and validation splits, with original text columns removed so that only tensor fields like input_ids and attention_mask remain in the final dataset objects passed to the training loop.

D. *PEFT Training and GPU Configuration*

i. Training Environment

The training leveraged a Google Compute Engine backend equipped with a single 80 GB of dedicated GPU RAM, enabling efficient training of the 4-bit quantized Mistral-7B model with LoRA adapters.

ii. GPU Memory Optimization

Several memory optimization techniques were employed to maximize GPU utilization during training. Gradient checkpointing was enabled, trading computation for memory by recomputing activations during the backward pass rather than storing them, resulting in a 30-50% reduction in activation memory at the cost of approximately 20% slower training.

iii. Resource Utilization During Training

During active training, GPU memory utilization peaked at approximately 10-12 GB, representing just 12-15% of the available 80 GB VRAM. This substantial headroom prevented out-of-memory errors and allowed for potential batch size increases in future experiments. System RAM usage remained stable at 4-6 GB during training, well within the 167.1 GB available

E. Full Fine-Tuning: Hardware Feasibility Analysis

We assessed the computational feasibility of performing full fine-tuning (FFT) on the 7.24-billion parameter Mistral model using a single NVIDIA A100 GPU. The memory requirements for FFT are dominated not just by the static model weights, but also by the dynamic overhead of the optimization process. In standard mixed-precision training (BF16), the model parameters occupy approximately 14 GB. However, the standard AdamW optimizer requires maintaining two moment vectors per parameter in full FP32 precision to prevent numerical instability. These optimizer states consume an additional 56 GB, pushing the static memory requirement to 70 GB before a single forward pass is computed. Even with memory-efficient 8-bit optimizers, the combined state would exceed 40 GB, leaving no room for gradients or activations.

Furthermore, the training process requires storing gradients, another 14 GB in BF16, and intermediate activation maps proportional to the batch size and sequence length of 512 tokens. Summing these components (i) weights 14 GB, (ii) optimizer states 56 GB, and (iii) gradients 14 GB yields a theoretical minimum footprint of roughly 84 GB, which is more than the available VRAM. While techniques like gradient checkpointing can reduce activation memory, they cannot compress the fixed parameter and optimizer state requirements.

Table 4: Parameter Memory Requirement

States	Bytes per parameter (BF 16)	**Total Memory (7B parameter LLM)**
Weights	2 Bytes	**~14 GB**
Adam Optimizer	8 Bytes (FP32)	**~ 56 GB**
Gradients	2 Bytes	**~14 GB**
Activations	~2 (varies)	**~14 GB**
Total	**~ 14**	**~98 GB**

Considering these constraints, full fine-tuning a 7.2B parameter model on a single A100 is technically infeasible without significant offloading to system RAM, which would impose prohibitive latency penalties.

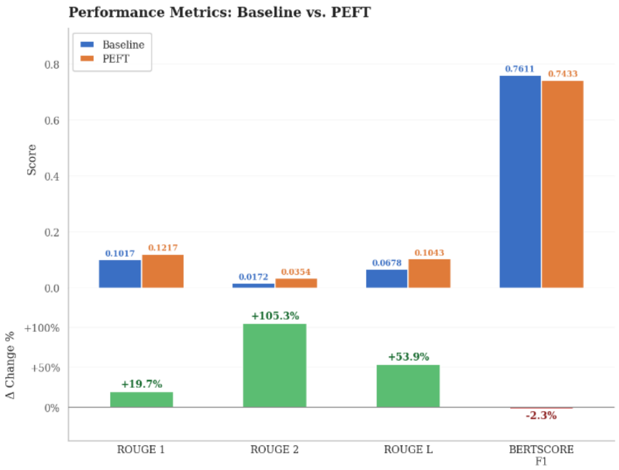
VII. RESULT

The parameter-efficient fine-tuning approach yielded mixed but generally positive results across the comprehensive evaluation metrics. The PEFT model demonstrated notable improvements in several key natural language generation metrics compared to the base Mistral-7B model. The most substantial improvement was observed in the ROUGE-2 score, which measures bigram overlap between generated and reference responses. The PEFT model achieved a ROUGE-2 score of 0.0354 compared to the base model's 0.0172, representing a remarkable 105.3% relative improvement. This substantial gain indicates that fine-tuning significantly enhanced the model's ability to generate coherent two-word sequences that align with reference medical responses. ROUGE-1 scores, measuring unigram overlap, showed a modest improvement from 0.1017 to 0.1217, a 19.7% increase. Similarly, ROUGE-L, which captures the longest common subsequence, improved by 53.9% from 0.0678 to 0.1043. These consistent improvements across all ROUGE variants suggest that the PEFT model generates responses with better lexical overlap with reference texts, indicating improved content relevance in medical dialogues. BERTScore F1, which evaluates semantic similarity using contextual embeddings, however, showed a slight decrease from 0.7611 to 0.7433, representing a 2.3% reduction

Table 5:Performance Metric of The Models

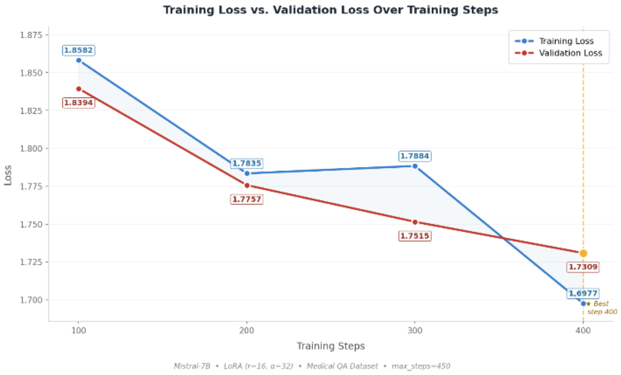
Metric	Baseline	PEFT	Change %
ROUGE 1	0.1017	0.1217	19.7
ROUGE 2	0.0172	0.0354	105.3
ROUGH L	0.0678	0.1043	53.9
BERTSCORE_F1	0.7611	0.7433	-2.3

Figure 2: Performance Metrics: Baseline vs. PEFT



The validation loss exhibits a smooth, monotonic decrease throughout training without significant oscillations or spikes, indicating stable optimization dynamics. However, as training progressed beyond 350 steps, the rate of improvement began to diminish considerably, with the loss curve showing clear signs of plateauing around 1.73. This plateauing behavior informed the decision to set the maximum training steps to 400, as additional computation would yield diminishing returns without meaningful improvements in model performance.

Figure 3: Loss Curves



VIII. DISCUSSION

The experimental results demonstrate that parameter-efficient fine-tuning using LoRA on a 4-bit quantized Mistral-7B model yields meaningful improvements in medical dialogue generation, particularly in lexical alignment metrics. The 105.3% improvement in ROUGE-2 represents the most significant gain, suggesting that the fine-tuned model has effectively learned domain-specific bigram patterns critical for medical communication. This finding aligns with the nature of medical dialogues, where standardized two-word phrases like "differential diagnosis," "adverse effects," and "vital signs" carry substantial clinical meaning and appear frequently in doctor-patient interactions.

IX. CONCLUSION

This study demonstrates that parameter-efficient fine-tuning with LoRA on a 4-bit quantized Mistral-7B model achieves meaningful improvements in medical dialogue generation, while requiring only 400-450 training steps and modest computational resources. The rapid convergence and clear loss plateau validate the efficiency of the PEFT approach for domain adaptation, while the mixed metric performance highlights the need for multifaceted evaluation frameworks that capture both lexical accuracy and clinical appropriateness.

X. ETHICAL, LEGAL AND PROFESSIONAL CONSIDERATIONS

The development and deployment of LLMs for medical dialogue generation carries significant ethical, legal, and professional responsibilities that must be carefully considered. The use of the MedDialog dataset raises concerns about patient privacy and informed consent, as the original conversations may have been collected without explicit permission for machine learning research, and despite anonymization efforts, the risk of re-identification persists, particularly in cases of rare medical conditions. Furthermore, the potential for harm through clinical hallucinations where the model generates plausible sounding, but factually incorrect medical information could lead to adverse patient outcomes if such outputs were relied upon without professional oversight. Responsible development requires ongoing interdisciplinary collaboration between AI researchers, clinicians, ethicists, and patient representatives to ensure that technical advancements align with patient welfare, professional standards, and societal values.

XI. REFERENCES

[1] B. Pawar, "AI in Healthcare Market Size, Share & Industry Analysis, By Platform (Solutions and Services), By Application (Robot-Assisted Surgery, Virtual Nursing Assistant, Administrative Workflow Assistance, Clinical Trials, Diagnostics, and Others), By End-user (H," Fortune Business Insights, 2026.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, Illia Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, 2017.

[3] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, "Improving Language Understanding by Generative Pre-Training," 2018.

[4] Devlin, J., Chang, M. W., Lee, K., Toutanova, K., "Bert: Pre-training of deep bidirectional transformers for language understanding," *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2019.

[5] Singhal, K., Tu, T., Gottweis, J., "Toward expert-level medical question answering with large language models," *Nature Medicine*, 2025.

[6] Hao Chen, Ye He, Yuchun Fan, Yukun Yan, Zhenghao Liu, Qingfu Zhu, Maosong Sun, Wanxiang Che, "Know More, Know Clearer: A Meta-Cognitive Framework for Knowledge Augmentation in Large Language Models," 2025.

[7] EJ Hu, Y Shen, P Wallis, Z Allen-Zhu, Y Li, S Wang, L Wang, W Chen, "Lora: Low-rank adaptation of large language models," 2022.

[8] Artsi, Y., Sorin, V., Glicksberg, B. S., Korfiatis, P., Freeman, R., Nadkarni, G. N., Klang, E., "Challenges of Implementing LLMs in Clinical Practice: Perspectives," *Journal of Clinical Medicine*, 2025.

[9] Xuehai He, Shu Chen, Zeqian Ju, Xiangyu Dong, Hongchao Fang, Sicheng Wang, Yue Yang, Jiaqi Zeng, Ruisi Zhang, Ruoyu Zhang, Meng Zhou, Penghui Zhu, Pengtao Xie, "MedDialog: Two Large-scale Medical Dialogue Datasets," 2020.

[10]

XII. APPENDICES

Project code
Task 1 Github url : https://github.com/0xschool/Medical-Consultation/blob/main/scripts/LLM_coursework.ipynb

Task 2 Github url : https://github.com/0xschool/Generative-AI-Augmentation