

Arabic Clinical Decision Support under a Limited GPU Budget: LoRA versus Full Fine-Tuning for Medical Question Answering

Ibrahim Abaker Hashem

University of Sharjah

Omar Elgendy

oelgendy@sharjah.ac.ae

University of Sharjah

Ali Bou Nassif

University of Sharjah

Ismail Shahin

University of Sharjah

Ashraf Elnagar

University of Sharjah

Ayad Turkey

University of Sharjah

Imad Afyouni

University of Sharjah

Research Article

Keywords: clinical decision support, healthcare question answering, Arabic medical NLP, large language models, parameter-efficient fine-tuning, GPU-constrained deployment

Posted Date: September 23rd, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10807173/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Arabic Clinical Decision Support under a Limited GPU Budget: LoRA versus Full Fine-Tuning for Medical Question Answering

Ibrahim Abaker Hashem¹, Omar Elgendy^{1*}, Ali Bou Nassif¹, Ismail Shahin¹,
Ashraf Elnagar¹, Ayad Turkey¹, Imad Afyouni¹

^{1*}University of Sharjah, Sharjah, United Arab Emirates.

*Corresponding author(s). E-mail(s): oelgendy@sharjah.ac.ae;

Contributing authors: ihashem@sharjah.ac.ae; anassif@sharjah.ac.ae; ismail@sharjah.ac.ae;
ashraf@sharjah.ac.ae; aturky@sharjah.ac.ae; iafyouni@sharjah.ac.ae;

Abstract

Teams deploying Arabic-language clinical question-answering assistants under a limited GPU budget face a consequential choice of open-weight base model and adaptation strategy. We compare low-rank adaptation (LoRA) with full fine-tuning for Arabic medical multiple-choice question answering, using a two-stage pipeline (domain adaptation on AraMed, task adaptation on MedAraBench) across four open-weight models spanning the Arabic pre-training spectrum: Jais-2-8B-Chat, Fanar-1-9B-Instruct, Llama-3.1-8B, and Qwen2.5-7B-Instruct, evaluated on 4,761 questions with three random seeds and bootstrap confidence intervals. Full fine-tuning outperforms default-configuration LoRA by 1.3–1.6 Macro-F1 points on the two original 8B models, but a LoRA rank-and-dropout sweep closes most of the gap, and 4-bit QLoRA matches full fine-tuning on a single 24 GB GPU. Arabic-centric pre-training confers a zero-shot advantage that largely vanishes after task fine-tuning. A well-configured low-rank adapter is therefore an adequate adaptation strategy for Arabic clinical question answering.

Keywords: clinical decision support, healthcare question answering, Arabic medical NLP, large language models, parameter-efficient fine-tuning, GPU-constrained deployment

1 Introduction

Large language models (LLMs) have shown promising results for different clinical applications involving natural language processing, such as triaging patients, answering medical questions, and diagnosing illnesses. In general, artificial intelligence (AI) has shown remarkable potential in clinical domains such as diagnosis aid and medical imaging [8, 16]. For instance, Med-PaLM 2 [22] achieved or outperformed physicians’ accuracy in standardized medical English exams, generating heightened interest in using LLMs for clinical decision-making assistance. Nevertheless, this research field is

heavily dominated by the English language. The medical LLM literature has scarcely utilized Arabic, a language that is spoken by more than 400 million people and is used for health care delivery in over 20 countries. Currently, there is no readily available language model for medical applications in Arabic because of the low availability of Arabic medical datasets and the complicated nature of the Arabic alphabet.

This gap matters in practice, and not only as a research question. A hospital group or a ministry of health in an Arabic-speaking country that

wants to field a clinical question answering assistant, whether for triage support, clinician reference, or patient guidance, does not start from a question about adapter ranks: it starts from an open-weight base model, a modest GPU allocation, and a delivery deadline. Figure 1 sketches that setting and the decision it forces. Which base model should the deployment start from, given that the available Arabic-capable models differ widely in how much Arabic they saw during pre-training? Should that model be adapted with a full-parameter update, which in our setting requires four 24 GB GPUs, or with a parameter-efficient adapter that fits on one? And is it worth spending part of a fixed budget on a domain-adaptation stage over naturalistic Arabic clinical text before task-specific training begins? The experiments reported in this paper are designed to answer those questions with numbers a practitioner can act on, and the same evidence speaks to the underlying scientific question of how the volume of Arabic pre-training in a base model interacts with the choice of adaptation strategy.

Fine-tuning is an integral step in adapting LLMs to specific languages and domains [24], but it offers a choice between two mainstream approaches. The first method, which provides the greatest flexibility but has significant computational costs and a catastrophic forgetting issue, involves updating every model parameter. Low-rank matrix decompositions are used in the second parameter-efficient tuning method to update a minimal fraction of the parameters [11], which can considerably reduce compute costs and maintain comparable performance levels [21]. There is currently no research on employing LoRA versus complete fine-tuning on Arabic-language medical question and answering (Q&A), even though parameter-efficient models have been thoroughly examined in English-language tasks. This disparity must be considered because, contrary to what English-specific results suggest, a different method of adaptation would be more appropriate in languages with rich morphology, limited pre-training data, and few domain-specific literature.

A significant, underexplored source of variation further complicates this matter. Arabic NLP open-weight base models exhibit large variability in the volume of Arabic encountered during pretraining. Arabic constitutes a small portion of the pretraining dataset for general-purpose models

such as Llama 3 [9], which are primarily trained on English. Conversely, Arabic-focused methods [20] use balanced bilingual training and bilingual Arabic tokenization to treat Arabic as a primary pretraining language. Regarding the relative efficacy of LoRA versus complete fine-tuning for downstream medical tasks, this variation in native Arabic pretraining may have important practical implications. Regardless of the language profile of the base model, parameter-efficient adaptation may be sufficient if the pretraining is sufficiently linguistically grounded (for example, Arabic-centric pretraining); otherwise, complete fine-tuning may be necessary.

In this paper, we systematically compare the LoRA method with complete fine-tuning for the task of Arabic medical question answering on general-purpose and Arabic-focused LLMs, and we frame the comparison throughout as the adaptation-strategy decision faced by a compute-constrained team deploying an Arabic clinical question answering assistant.

The contributions of this work are as follows:

- **A two-stage Arabic medical fine-tuning pipeline with Stage 1 diagnostics.** We provide a framework that divides task-specific fine-tuning on MedAraBench [1], a multi-specialty Arabic medical multiple-choice question (MCQ) benchmark, from domain adaptation (continuous pretraining on AraMed [2], a 270K-pair Arabic medical corpus of which we use the 109,834 question–relevant-answer pairs in its training split). To the best of our knowledge, this is the first study to separate Arabic medical question answering (QA) adaptation into these two distinct phases. We also offer a diagnostic analysis of the reasons why the AraMed domain-adaptation stage does not consistently produce a gain over direct task fine-tuning at this scale.
- **A factorial LoRA-versus-full comparison extended to four models and to LoRA and QLoRA configurations** We test all four possible combinations of LoRA and full fine-tuning in both phases of training on two matched 8B-parameter base models on either end of the spectrum of Arabic pretraining: the English-biased Llama-3.1-8B and the Arabic-focused Jais-2-8B-Chat. This guarantees that pretraining language, not model size, is the cause of any observed effects. We then extend the comparison

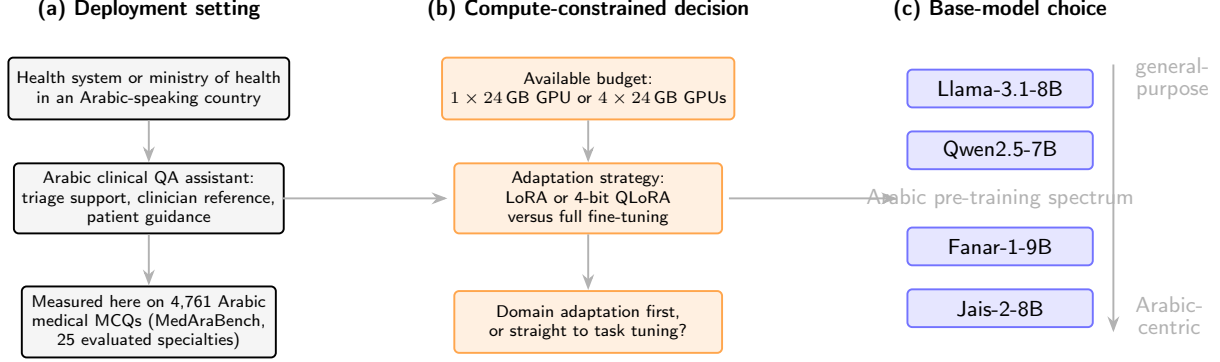


Fig. 1 The application setting that motivates this study. (a) A health system or ministry of health in an Arabic-speaking country wants to field an Arabic clinical question answering assistant; in this paper its competence is measured on 4,761 native Arabic medical multiple-choice questions spanning the 25 fine-grained specialties retained for evaluation. (b) A team working under a limited GPU budget must decide how to adapt an open-weight model, choosing between a parameter-efficient adapter that trains on a single 24 GB GPU and a full-parameter update that requires four of them, and deciding whether to insert a domain-adaptation stage before task-specific tuning. (c) The candidate base models sit at different points on the Arabic pre-training spectrum, from the general-purpose Llama-3.1-8B and Qwen2.5-7B to the Arabic-centric Fanar-1-9B and Jais-2-8B. The training pipeline that implements these choices is given in Figure 2.

to two additional 7–9B models: the general-purpose Qwen2.5-7B-Instruct and the Arabic-centric Fanar-1-9B-Instruct. To ensure that the LoRA setting is well-tuned rather than assumed, we supplement the default-configuration contrast with a LoRA hyperparameter sweep over rank, learning rate, dropout, and target modules, along with 4-bit QLoRA conditions.

- **The first evidence on how Arabic-centric pretraining interacts with the adaptation strategy.** We offer the first rigorous quantification of the extent to which the decision of fine-tuning either fully or parameter-efficiently is influenced by the extent of Arabic pretraining of the base language model. Our findings indicate that the order of adaptation methods is mostly insensitive to the pretraining language, and the 6.5-point zero-shot benefit of Arabic-specific pretraining almost entirely disappears after direct fine-tuning. We base these conclusions on a reliability analysis that reruns the key configurations with three random seeds and reports 95% bootstrap confidence intervals and McNemar’s test, showing that the small full-versus-LoRA differences are largely within seed-level variation.

In our experiments, full fine-tuning holds a small and qualified advantage over a default-configuration LoRA at the task-adaptation stage for the two original 8B base models, and the relative ranking of the adaptation strategies is largely unaffected by the pretraining language profile.

Despite Jais-2-8B’s 6.5 point zero-shot advantage from Arabic-centric pretraining, that gap narrows to less than one point after fine-tuning. For the two original 8B models, our results indicate that the adaptation outcome is driven more by target-task complexity than by the pretraining language of the base model. This preference for full fine-tuning is, however, modest in size and sensitive to the LoRA configuration, and it does not extend uniformly to the additional model families we test; therefore, we frame full fine-tuning as a reasonable default only when compute permits, rather than a categorical recommendation.

The remainder of this paper is organized as follows. Section 2 presents the background and motivation across four axes: medical LLMs, medical QA benchmarks and datasets, Arabic LLMs, and parameter-efficient fine-tuning methods. Section 3 develops the theoretical motivation for why the extent of Arabic pretraining might interact with the choice of adaptation strategy. Section 4 describes the experimental setup, detailing the base models, the two-stage fine-tuning pipeline, dataset preprocessing, hyperparameter configuration, and evaluation protocol. Section 5 presents and analyzes the results across all 22 experimental conditions, examining the effects of fine-tuning strategy, domain adaptation, and Arabic-centric pretraining on downstream performance. Section 6 concludes with a summary of findings and directions for future work.

2 Background and Motivation

2.1 Medical Large Language Models

Over the past few years, there has been a steady and quick advancement in the use of large language models for clinical question answering. Early domain-adapted models trained on biomedical corpora such as PubMed suggested that targeted pretraining might greatly enhance performance on structured benchmarks such as MedQA [12] and MedMCQA [17]. However, their gains remained modest in comparison to what general-purpose scale would eventually enable. The creation of large instruction-tuned models dramatically changed this course. By combining a stronger base model with medical domain fine-tuning and retrieval-augmented reasoning, [22] demonstrated performance on USMLE-style questions that, according to physician review, approached or exceeded generalist responses across several clinical axes. This study offered two innovative inference strategies: chain-of-retrieval, a sequential pipeline that bases replies in externally retrieved sources, and ensemble refinement, which is based on Google’s PaLM 2 backbone and combines various reasoning routes before committing to a final solution. The final model outperformed the original MedPaLM by more than 19 percentage points, scoring 86.5% on USMLE-style MedQA [12] questions. In human evaluations, doctors favored its long-form responses above those of other doctors on eight out of nine clinical axes. When considered collectively, these findings set a clear precedent: domain fine-tuning on top of a strong general-purpose backbone is a scalable and efficient route to clinically meaningful LLMs. They also revealed a more fundamental conflict: benchmark accuracy is not enough for clinical implementation on its own. Evaluation has to become increasingly complex and specialized rather than just bigger, with alignment-aware criteria at its core, as models get closer to physician-level performance on structured exams.

MedAlpaca [10] adopts an alternative strategy, putting accessibility and deployability ahead of scalability. Medical Meadow, a 160k+ entry instruction-tuning dataset collected from medical flashcards, Stack Exchange forums, WikiDoc, and well-known benchmarks like MedQA and CORD-19, is its main contribution. When LLaMA (7B

and 13B) is fine-tuned using this data and evaluated across USMLE Steps 1-3, the refined models consistently perform better than their basic counterparts, with MedAlpaca-13B achieving 0.602 on Step 3. One of the main motivators for the project is patient privacy. The authors argue that the use of locally runnable open-source alternatives is necessary since relying on closed, API-based models, like GPT [4] and its counterparts, is essentially incompatible with clinical deployment. They investigate LoRA and 8-bit quantization for effective fine-tuning at a quantifiable performance penalty in order to further reduce the processing barrier. The writers also freely acknowledge that hallucinations are a basic flaw, which they accurately note is particularly dangerous in medical contexts.

A different line of work addressed a need on the generative side of biological NLP in parallel with these fine-tuning-focused initiatives. While generative modeling of biological literature has received relatively little attention, BERT-style models, such as BioBERT and PubMedBERT, have been extensively studied for understanding tasks. Microsoft’s approach to this gap is BioGPT [14], a GPT-2 architecture pretrained from scratch on 15M PubMed abstracts using a domain-specific vocabulary. The outcomes are impressive: top performance on three biomedical relation extraction benchmarks and state-of-the-art on PubMedQA (78.2%). One important discovery is that significant increases in specialized domains are produced by in-domain pretraining from scratch as opposed to ongoing pretraining over a general-purpose vocabulary. The authors also demonstrate that improved generalization results from producing outputs in natural language as opposed to structured tokens with unique markers. This is further extended by the larger BioGPT-Large (1.5B parameters), which raises PubMedQA performance to 81.0%.

Building on this pretraining understanding, PMC-LLaMA [25] fills the instruction-following vacuum created by models such as BioGPT. By merging two training phases on top of LLaMA, PMC-LLaMA is a methodical effort to create a fully open-source, medically-adapted LLM. In order to create a richer embedding space for clinical terminology prior to any task-specific training, the initial step is data-centric knowledge injection, which involves ongoing pretraining on 4.8

million biomedical papers and 30 thousand medical textbooks (MedC-K). The second step involves fine-tuning medical instruction on a 202M-token dataset (MedC-I) that includes three different sources of data: knowledge graph prompting from UMLS, rationale-augmented QA (produced with ChatGPT to add chain-of-thought explanations to existing benchmarks), and doctor-patient chats. In terms of efficiency, the resulting 13B-parameter model performs better on MedQA, MedMCQA, and PubMedQA than ChatGPT (175B). Their ablations clearly demonstrate that both phases are important on their own: information injection by itself is helpful, instruction tweaking by itself is more beneficial, but combining them with a variety of data types-particularly adding rationale and knowledge graphs-produces the best outcomes. This body of work’s almost exclusive emphasis on English is a significant drawback. The evaluation frameworks are predicated on an anglophone clinical setting, and MedQA, MedMCQA, PubMedQA, and the datasets used to train these models are primarily English-language resources. For most patient groups globally, where professional communication takes place in other languages-Arabic being one of the most widely spoken languages-this creates a significant divide. The degree to which the training and architectural ideas gained in this field of work translate to non-English medical settings is still mostly unknown.

2.2 Medical QA Benchmarks and Datasets

A three-step, standardized test called the United States Medical Licensing Examination (USMLE) is used to gauge a doctor’s clinical ability before granting them a license in the US. Basic biomedical science information is tested in Step 1, clinical knowledge and skills necessary for supervised patient care are evaluated in Step 2, and the capacity to use medical knowledge in unsupervised clinical situations is assessed in Step 3. USMLE has become the most extensively used benchmark for assessing medical AI systems because of its strict design and widespread acceptance as a gauge of physician-level thinking. It is also the basis for a number of well-known NLP benchmarks.

MedQA [12] presented the first extensive free-form multiple-choice open-domain question answering dataset derived from professional medical

board exams, building directly upon the USMLE framework. With 12,723, 34,251, and 14,123 questions in each of the three languages-English, simplified Chinese, and traditional Chinese-the English part of the collection was created only from USMLE Steps 1-3 materials. With 98% of English questions presenting complete patient case scenarios rather than isolated factual queries, MedQA requires models to perform multi-hop clinical reasoning based on medical textbook knowledge, in contrast to previous medical QA datasets that relied on retrieval-based or span-extraction formats. The authors created a document collection of eighteen English medical textbooks totaling more than 12.7 million tokens to bolster this claim. Even the best models at the time of publishing, such as BioBERT-Large, only obtained 36.7% accuracy on the English subset, according to baseline studies, highlighting the depth of reasoning required by the benchmark. Since then, MedQA has emerged as the preeminent standard for assessing medical LLMs, acting as the main evaluation objective across the corpus of work covered in this section.

By utilizing two Indian postgraduate medical entrance exams, the All India Institute of Medical Sciences (AIIMS PG) and the National Eligibility cum Entrance Test (NEET PG), MedMCQA [17] expands the multiple-choice medical QA paradigm to a significantly larger and more diverse examination setting. The dataset includes 193,155 high-quality multiple-choice questions (MCQs) covering 2,400 healthcare topics in 21 medical subjects. Each sample includes not only a question, candidate options, and the right answer, but also a thorough natural language explanation of the solution, a feature that is lacking in the majority of contemporaneous benchmarks. The majority of the questions are open-ended and non-factoid, requiring models to practice more than ten different reasoning skills, such as multi-hop reasoning, diagnosis, therapy selection, and mathematical inference. The dataset is divided by exam rather than by question to reflect the generalization requirements of real-world examination settings: the training set is made up of mock and online test series, the development set is made up of NEET PG questions, and the test set is made up of AIIMS PG questions from 1991 to the present. This design decision minimizes cross-split leakage of nearly identical questions. Even domain-adapted models fall well short of the

expert-level competency these tests are intended to assess, as demonstrated by baseline evaluations using domain-specific transformer models, such as BioBERT and PubMedBERT, which yield a best accuracy of 47% on the test set, far below the estimated human expert performance of 90%.

By basing evaluation on biomedical research literature rather than clinical examination materials, PubMedQA [13] adopts a radically different way to addressing medical questions. The questions are taken straight from the titles of PubMed articles that are written as research questions, with the conclusion acting as a long-form response and the associated structured abstract-aside from its conclusion-serving as the context. In order to reflect the conditional and frequently ambiguous nature of findings in empirical biomedical research, models must respond to each research question with one of three labels: *yes*, *no*, or *maybe*. The dataset is divided into three subsets: 211,300 artificially generated examples (PQA-A) created by turning statement-form article titles into questions, 61,200 unlabeled instances (PQA-U), and 1,000 expert-annotated instances (PQA-L). PubMedQA’s emphasis on quantitative reasoning sets it apart from previous biomedical QA standards. Rather than collecting discrete facts from text, almost all cases require models to interpret numerical data, such as effect sizes, p-values, and group comparisons. On the expert-annotated test set, baseline tests employing multi-phase BioBERT fine-tuning achieve 68.1% accuracy compared to a single-human performance of 78.0%, leaving a significant gap that spurs more scientific reasoning research.

Recent research has pushed for more clinically realistic evaluation frameworks, going beyond multiple-choice formats. Inspired by the Objective Structured Clinical Examination (OSCE), which is used in medical education, MedQA-CS [26] presents an AI-Structured Clinical Examination (AI-SCE) framework. Instead of using multiple-choice questions to test clinical knowledge, MedQA-CS assesses LLMs at Miller’s Pyramid’s “shows how” level, evaluating actual clinical skills in four areas: information gathering, physical examination, closure, and differential diagnosis. The benchmark uses a dual-role evaluation pipeline with one LLM acting as a medical student (Med-StuLLM) and another as a clinical skills examiner

(MedExamLLM). It consists of 1,667 instruction-based data points that are derived from 44 USMLE Step 2 CS cases. According to experimental studies, state-of-the-art models score significantly lower on MedQA-CS, indicating a persisting gap between clinical knowledge and clinical competence, even while they reach or surpass physician-level performance on knowledge-based benchmarks like MedQA.

Despite the breadth and rigor of these benchmarks, they share a critical limitation: with the exception of MedQA’s Chinese subsets, the medical QA evaluation landscape remains overwhelmingly English-centric, leaving the development and assessment of medical AI systems in other languages-Arabic in particular-severely underserved.

In contrast to previous work that relied on machine translation of English resources, MedAraBench [1] is the first large-scale natively Arabic medical multiple-choice question benchmark. It was created by manually digitizing scanned paper-based examinations sourced from regional Arabic medical schools. The dataset, which includes 24,883 multiple-choice questions (MCQs) covering 19 top-level medical disciplines and five difficulty levels corresponding to years of medical school (Y1–Y5), is divided into 19,894 training and 4,989 test samples using stratified random splitting after a thorough filtering and quality control pipeline. These totals match the counts reported in the original MedAraBench paper; the publicly released version we download differs slightly, providing 19,891 training and 4,959 test samples, which are the raw counts used throughout our experiments. Specialty information is annotated at two levels of granularity: the 19 top-level disciplines quoted above, and a finer partition of 26 specialty labels nested beneath them. The coarse level is used when describing the corpus, and the fine-grained labels are the unit of the per-specialty analysis in Section 5.2, where one label is dropped for having too few test items, leaving 25. Two complementary methods were used to evaluate the dataset quality: an LLM-as-a-judge protocol in which GPT-o3, Gemini-2.0-Flash, and Claude-Sonnet-4 evaluated the entire test set using the same criteria, and expert human evaluation by two board-certified clinicians, which produced slight-to-fair inter-annotator agreement across all

metrics (Cohen’s κ ranging from 0.152 to 0.555). A significant performance gap between model categories is revealed by benchmarking experiments across 16 proprietary and open-source models in a zero-shot setting: proprietary reasoning models like GPT-o3 and GPT-5 achieve the highest accuracies at 76.5%, while Arabic-centric open-source models like ALLaM (44.7%) and Fanar (49.8%) lag significantly behind, with no model approaching expert-level performance. The significance of focused training on Arabic medical data is highlighted by parameter-efficient fine-tuning tests using QLoRA on Llama-3.1-8B, which further show that domain-specific adaptation produces significant gains, increasing accuracy from 17.0% to 32.0%—an 88% relative improvement. However, despite its size, MedAraBench has a number of basic drawbacks that prevent it from being used directly as a fine-tuning resource in this study. Deduplication by itself cannot rule out the preservation of semantically comparable questions with slight surface-level variation. Approximately 20% of the training split consists of duplicate or near-duplicate questions that can be linked to the replication of exam materials across source sources. Because this figure includes semantically equivalent items that differ only in surface form, and which exact-match deduplication cannot detect, it is considerably larger than the proportion that our own cleaning pipeline is able to remove in total (2,253 of 19,891 training items, or 11.3%). A non-trivial subset of questions have multi-label answer keys, like "D+E" or "A+B+C+D+E," which make them structurally incompatible with single-answer multiple-choice questions (MCQ) fine-tuning. Additionally, a portion of the questions contain free-text Arabic strings and answer keys that are distorted due to irreversible digitization artifacts from the original scanned handwritten exams. Lastly, the difficulty distribution is strongly biased toward first-year medical content, which accounts for more than 60% of the training split. This distributional bias runs the risk of creating models that are more focused on basic factual recall than the higher-order clinical reasoning required by Arabic medical question answering. The use of AraMed [2] as the domain adaptation resource for the fine-tuning experiments in this work is motivated by these constraints taken together.

AraMed [2] addresses the scarcity of Arabic medical QA resources by introducing a large-scale

corpus of 270,000 question-answer pairs sourced from Altibbi ¹, a widely used Arabic medical consultation platform in the MENA region. Questions were collected from user submissions spanning 2020 to 2021, covering 85 medical categories ranging from gynecological diseases and general medicine to pharmacology and cardiovascular conditions, with an average question length of 23.9 tokens and an average answer length of 34.2 tokens. Each question is paired with a relevant doctor-provided answer and an automatically generated irrelevant answer through a rule-based sliding window annotation approach, framing the corpus as a binary answer selection task. The dataset is primarily composed of Modern Standard Arabic, reflecting the platform’s commitment to standardized medical communication, and undergoes a preprocessing pipeline including diacritic removal, text normalization, and elimination of non-textual content. Baseline experiments using Arabic-specific transformer models demonstrate strong performance on the answer selection task, with AraBERTv2 achieving an F1 score of 96.73%, establishing a solid benchmark for future Arabic medical NLP work. In this work, we leverage AraMed as a domain adaptation resource in the first stage of our fine-tuning pipeline, exploiting its scale and linguistic diversity to expose the model to naturalistic Arabic medical language prior to task-specific adaptation on structured clinical MCQs.

2.3 Base Language Models for Arabic NLP

An important turning point in Arabic generative NLP was reached with AraGPT2 [3], the first transformer-based Arabic language generation model trained from scratch. AraGPT2, which was created at the American University of Beirut, is pretrained on 77GB of Arabic text, totaling 8.8 billion words, from the OSCAR corpus, Arabic Wikipedia, the 1.5 billion word Arabic corpus, the OSIAN corpus, and news articles from the As-Safir newspaper. It is based on the GPT-2 causal language modeling architecture. Four size variations of the model are available, ranging from 135 million parameters (base) to 1.46 billion parameters (mega). The largest form uses the

¹<https://altibbi.com/>

Adafactor optimizer and the GROVER architecture to handle memory restrictions at scale. Human evaluation shows that AraGPT2-mega successfully tricked about 60% of Arabic-speaking evaluators into misclassifying machine-generated news articles as human-written, and evaluation on held-out Wikipedia articles yields a perplexity of 29.8 for the mega variant. This led the authors to release an accompanying AraELECTRA-based discriminator that achieved 98.7% F1-score on long-text detection. The model’s emergent factual knowledge and the limitations of prompt-based inference without fine-tuning are both evident in the zero-shot evaluation on Arabic question answering benchmarks, which shows a 24.6% manual accuracy. AraGPT2 demonstrated the feasibility of large-scale Arabic generative pretraining and formed the foundation for later Arabic-centric LLMs, even though it was not intended for instruction following or domain-specific tasks.

The Mohamed bin Zayed University of Artificial Intelligence (MBZUAI) and Inception collaborated to create Jais [20], an Arabic-centric decoder-only large language model that is the first open generative LLM created especially to give priority to Arabic above English. Jais, which is based on the GPT-3 architecture with 13 billion parameters, uses a number of technical enhancements, including as the SwiGLU activation function, ALiBi positional encodings for extended context generalization, and a unique bilingual tokenizer that gives equal weight to Arabic and English. The model is pretrained on 395 billion tokens, which include 232 billion English tokens, 116 billion Arabic tokens (derived from the largest Arabic corpus at the time at 72 billion distinct tokens, repeated 1.6 times), and the remaining source code. In contrast to massively multilingual models like BLOOM, Jais limits pretraining to Arabic and English only, with Arabic making up 33% of the Arabic–English portion of the mixture (116 of the 348 billion Arabic and English tokens), which corresponds to 29% of the full 395-billion-token corpus once source code is included. This intentional decision results in strong Arabic fluency and cultural awareness while maintaining the reasoning skills transferred from English. Over 3.6 million Arabic and 6 million English instruction-response pairs with extra safety alignment are used to further build an instruction-tuned version, Jais-chat. Building on this foundation, the

same team released Jais 2² in December 2025. It has a completely redesigned architecture, the largest Arabic-first training dataset to date, and significantly improved reasoning and fluency across Modern Standard Arabic and regional dialects. Jais 2 comes in two parameter variants: 8B and 70B. Because of its direct scale comparability with Llama-3.1-8B, the 8B variant was used as the Arabic-centric basis model in our fine-tuning tests.

Meta AI created the Llama 3 [9] family of open-weight foundation models, which are intended to natively enable multilinguality, coding, reasoning, and tool usage within a dense Transformer architecture. The largest model in the family has 405 billion parameters and a context window of up to 128K tokens. The 8 billion parameter variant, Llama-3.1-8B, offers a highly capable and computationally efficient option that has become the standard base model for fine-tuning research in the NLP community. Llama 3 models achieve performance comparable to top proprietary models like GPT-4 on a wide range of benchmarks after being pretrained on a sizable multilingual corpus and post-trained through supervised fine-tuning and reinforcement learning using human feedback. Llama 3’s multilingual exposure and robust reasoning abilities have made it a widely used baseline for cross-lingual transfer experiments, including Arabic NLP tasks, despite its pretraining emphasis being predominantly English-centric. In this work, Llama-3.1-8B is used as the general-purpose base model in our comparative fine-tuning study. It offers a reproducible and well-documented counterpart to the Arabic-centric Jais-2-8B and allows a controlled comparison of the interaction between the degree of Arabic pretraining in the base model and the fine-tuning strategy. To broaden the range of Arabic pretraining represented in our comparison, we additionally consider two more recent open-weight models. Fanar-1-9B-Instruct is drawn from the Fanar platform [23], an Arabic-centric generative system developed in Qatar whose flagship language models are trained with a strong emphasis on Modern Standard Arabic, regional dialects, and culturally grounded content. Its instruction-tuned nine-billion-parameter variant is positioned as a general-purpose Arabic assistant,

²<https://huggingface.co/inceptionai/jais-2-8b>

which makes it a useful second point on the Arabic-centric end of the pretraining spectrum alongside Jais-2-8B.

Finally, Qwen2.5-7B-Instruct [19] represents the general-purpose, multilingual end of the spectrum. The Qwen2.5 series is pretrained on a large and heterogeneous corpus of roughly eighteen trillion tokens with broad multilingual coverage but no particular Arabic emphasis, and its seven-billion-parameter instruction-tuned model is a strong and widely used baseline at a scale close to the other models considered here. It therefore serves as a second general-purpose comparison point alongside Llama-3.1-8B, allowing us to test whether the behaviour observed for Llama generalizes to another competitive non-Arabic-centric model.

Even though Arabic-centric and strong multilingual LLMs are becoming more widely available, and now span a spectrum from heavily Arabic-focused models such as Jais-2-8B and Fanar-1-9B to general-purpose models such as Llama-3.1-8B and Qwen2.5-7B, no previous research has thoroughly investigated whether the choice of fine-tuning strategy—full fine-tuning versus parameter-efficient adaptation—interacts differently with Arabic-centric and general-purpose base models in the context of answering Arabic medical questions. This gap, across a broader and more representative set of base models than any single pair, serves as a direct motivator for this work’s comparative experimental design.

2.4 Fine-tuning and Parameter-Efficient Adaptation

Full fine-tuning, which modifies all of the model’s parameters on a downstream task dataset, allows a pretrained model to maximally adapt its internal representations to the target domain. Because it does not restrict which weights can change or how much, this technique has repeatedly shown high performance across NLP tasks, allowing the model to undergo significant representational changes in response to new input. However, full fine-tuning has become more expensive as pretrained language models have grown to billions of parameters. This is due to the fact that updating all weights requires storing optimizer states and full-precision gradients for every parameter, resulting in memory needs that scale linearly with model size. When several task-specific models are needed at the same

time, full fine-tuning for models like Llama-3.1-8B and Jais-2-8B requires large GPU memory and compute budgets and creates independent model copies for each fine-tuned version, resulting in significant storage overhead. To address these limitations, a family of parameter-efficient fine-tuning (PEFT) methods has been developed with the goal of achieving performance comparable to full fine-tuning while updating only a small fraction of model parameters. Among these, Low-Rank Adaptation (LoRA) [11] has emerged as the dominant approach. LoRA is motivated by the observation that weight updates during model adaptation have a low intrinsic rank. For a pretrained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA constrains its update as $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$, and $r \ll \min(d, k)$ [11]. During training, W_0 remains frozen while only A and B are updated, dramatically reducing the number of trainable parameters. Applied to GPT-3 175B, LoRA reduces trainable parameters by a factor of 10,000 and GPU memory requirements by a factor of 3, while matching or exceeding full fine-tuning performance on RoBERTa, DeBERTa, GPT-2, and GPT-3 across multiple benchmarks and crucially, introducing no additional inference latency since the adapter matrices can be merged with the original weights at deployment time. By combining LoRA with 4-bit quantization of the underlying model, QLoRA [6] allows for the fine-tuning of models that would otherwise be impossible to train even with regular LoRA. Three major innovations are introduced by QLoRA, which backpropagates gradients through a frozen 4-bit quantized pretrained model into LoRA adapters: a novel 4-bit NormalFloat (NF4) data type that is information-theoretically optimal for normally distributed neural network weights; double quantization, which further reduces the memory footprint by quantizing the quantization constants themselves; and paged optimizers to handle memory spikes during training. Together, these advancements allow full 16-bit fine-tuning performance while fine-tuning a 65 billion parameter model on a single 48GB GPU. After just 24 hours of training on a single GPU, the resulting Guanaco model family, which was refined from LLaMA using QLoRA, achieves 99.3% of ChatGPT performance on the Vicuna benchmark. This shows that 4-bit quantization combined with low-rank adaptation imposes

negligible quality degradation relative to full precision training. Since then, QLoRA has emerged as the go-to method for domain-specific adaptation of big models in environments with limited resources, such as Arabic medical NLP, as the MedAraBench fine-tuning studies have shown.

Despite the widespread adoption of LoRA and QLoRA across NLP domains, no prior work has conducted a systematic comparison of LoRA versus full fine-tuning specifically for Arabic medical question answering, nor examined how this choice interacts with the degree of Arabic-centric pretraining in the base model. This gap constitutes the central contribution of this work.

3 Theoretical Motivation

The primary issue this paper addresses is whether the decision between full fine-tuning and parameter-efficient tuning depends on the amount of exposure the base model had to the Arabic language during its pretraining process. Two conflicting theories, based on proven facts, guide the experiment.

The low-rank sufficiency view

LoRA is based on the idea that the weight updates necessary for adapting a pretrained language model to a downstream task have an inherent low-rank structure, as noted by [11]. This means that only a limited number of pathways exist for the weight adjustments necessary to fully adapt a model. If this characteristic is applicable to Arabic medical QA, then updating the parameters with a rank-16 adaptation should effectively recover almost all of the benefits made by the full-parameter update. In other words, the two adaptations should be similar in performance. According to this approach, the level of Arabic-oriented pretraining is irrelevant when choosing the adaptation strategy, because the adaptation itself is inherently low-rank.

The distribution-shift view

One alternative argument is that the extent of distribution shift between pretraining and target task raises the intrinsic rank of the update needed. Arabic medical multiple-choice QA is simultaneously out-of-distribution on two axes: language (Modern Standard Arabic clinical text, morphologically rich and underrepresented in most pretraining corpora)

and task structure (five-way MCQ evaluated using token log-probability). A full-parameter update is anticipated to beat LoRA as a result of this combination change, making the rank-16 budget a binding limitation. Notably, this hypothesis predicts an interaction with pretraining language: the penalty to utilize LoRA should be larger for Llama than for Jais because the English-focused Llama-3.1-8B has a larger transition to Arabic clinical text than the Arabic-centered Jais-2-8B. Pretraining on native Arabic could reduce the effective distribution shift, rendering parameter-efficient adaptation sufficient for Jais even if it is insufficient for Llama.

What the experiment adjudicates

These two views yield different predictions about the same factorial design. The low-rank sufficiency view predicts no method gap and no interaction with pretraining language. The distribution-shift view predicts that full fine-tuning outperforms LoRA, and the size of this advantage depends on the base model’s exposure to Arabic. By comparing all four combinations of LoRA and full fine-tuning across two base models matched at 8B parameters but placed at opposite ends of the Arabic-pretraining spectrum, our design directly tests which model better describes adaptation behavior in this setting. A separate question concerns the two-stage structure itself: whether an intermediate domain-adaptation stage on open-ended Arabic medical text (AraMed) transfers usefully to structured multiple-choice reasoning, or whether the difference in format between the two stages limits or even reverses its benefit. Section 5 reports the results that resolve the disagreement between these hypotheses.

4 Methodology

4.1 Experimental Setup

In this study, we present a two-stage fine-tuning framework that separates the effects of base model pretraining and domain adaptation method on answering medical questions in Arabic. The framework is divided into two stages: (1) domain adaptation on a large-scale Arabic medical corpus to expose the model to the medical domain; and (2) task-specific fine-tuning on a multiple-choice question answering benchmark. LoRA [11] or full fine-tuning is used to train the model at each stage,

resulting in four main combinations for each base model: LoRA followed by LoRA, LoRA followed by full fine-tuning, full fine-tuning followed by LoRA, and full fine-tuning followed by full fine-tuning. An overview of the pipeline architecture is shown in Figure 2.

We chose two primary base models at a matching parameter count of about eight billion parameters as the controlled core of the study. These two models were chosen to reflect the extremes of the Arabic pretraining continuum, and they are complemented by two additional 7–9B models introduced later in this subsection. One of the most popular LLMs in NLP research is Llama-3.1-8B [9], a general-purpose model mostly focused on English. Despite lacking Arabic-focused pretraining, its extensive use in Arabic NLP benchmarking makes it an ideal choice for assessing whether downstream adaptation alone can compensate for the lack of exposure to native Arabic during pretraining. The second, Jais-2-8B, is the most well-known open Arabic-focused large language model to date and is the successor to Jais [20]. Jais-2-8B, which was developed with Arabic as a primary pretraining language, is Llama-3.1-8B’s natural experimental counterpart. This allows for a controlled assessment of whether native Arabic pretraining reduces the marginal advantage of full fine-tuning relative to parameter-efficient alternatives.

These two models of similar size and under the same experimental conditions provide a direct examination of the main study question: whether the extent of Arabic-focused pretraining of the base model interacts with the choice of fine-tuning method for downstream Arabic medical QA performance. Across all base models, the main results table comprises a total of 22 experimental conditions. For the two original 8B models, the design core is a 2×2 factorial over Stage 1 and Stage 2 approaches (LoRA or complete fine-tuning), resulting in eight two-stage conditions, together with four Stage-2-only baselines (LoRA or full fine-tuning without Stage 1) and two zero-shot baselines, for fourteen conditions in total. To test whether the observed pattern generalizes beyond Jais-2-8B and Llama-3.1-8B, we add two further open-weight models in the same 7–9B band, namely, the Arabic-centric Fanar-1-9B-Instruct and the general-purpose Qwen2.5-7B-Instruct, each evaluated zero-shot and under Stage-2-only

LoRA and full fine-tuning, resulting in six conditions. Finally, we add two 4-bit QLoRA conditions for Llama-3.1-8B and Jais-2-8B. Beyond these 22 conditions, to assess the reliability of the small full-versus-LoRA differences, we rerun the four key Stage-2-only configurations (LoRA and full fine-tuning for both original models) with three random seeds and report a LoRA hyperparameter sweep over rank, learning rate, dropout, and target modules; these reruns and sweep points are reported in dedicated tables rather than counted among the 22 conditions.

All experiments were conducted on an AWS EC2 g5.12xlarge instance, equipped with 4 NVIDIA A10G GPUs, which provide 96 GiB of GPU memory, 48 vCPUs, and 192 GiB of system RAM. The A10G GPUs operate at a compute capability 8.6, supporting bfloat16 mixed-precision training. The 8B-scale experiments run entirely on this instance.

4.2 Datasets and Preprocessing

Two separate Arabic medical datasets are used in the experimental workflow, with each serving a particular purpose. The final dataset sizes following preprocessing are summarized in Table 1. Stage 1 comprises Domain Adaptation (AraMed). AraMed, an open-ended Arabic medical question answering corpus, is used in the first stage. Each entry matches a patient query with a validated doctor response in free-text format. Because Stage 1 is continued pretraining on naturalistic clinical language rather than answer-selection training, we do not use the corpus in the binary form in which it is distributed: of the roughly 270K question–answer pairs described in Section 2.2, we retain only the question–relevant-answer pairs, discarding the automatically generated irrelevant answers, and we draw exclusively on the released training split. This yields the 109,834 usable question–answer pairs from which the Stage 1 counts reported below and in Table 1 are derived. AraMed is purposefully open-ended, in contrast to multiple-choice benchmarks, which makes it ideal for ongoing pretraining intended to introduce extensive medical domain knowledge into the base model before task-specific adaption. By design, minimal preprocessing occurs at this point. The question field is created by favoring the whole question description and reverting to the question title when the former is missing. Any

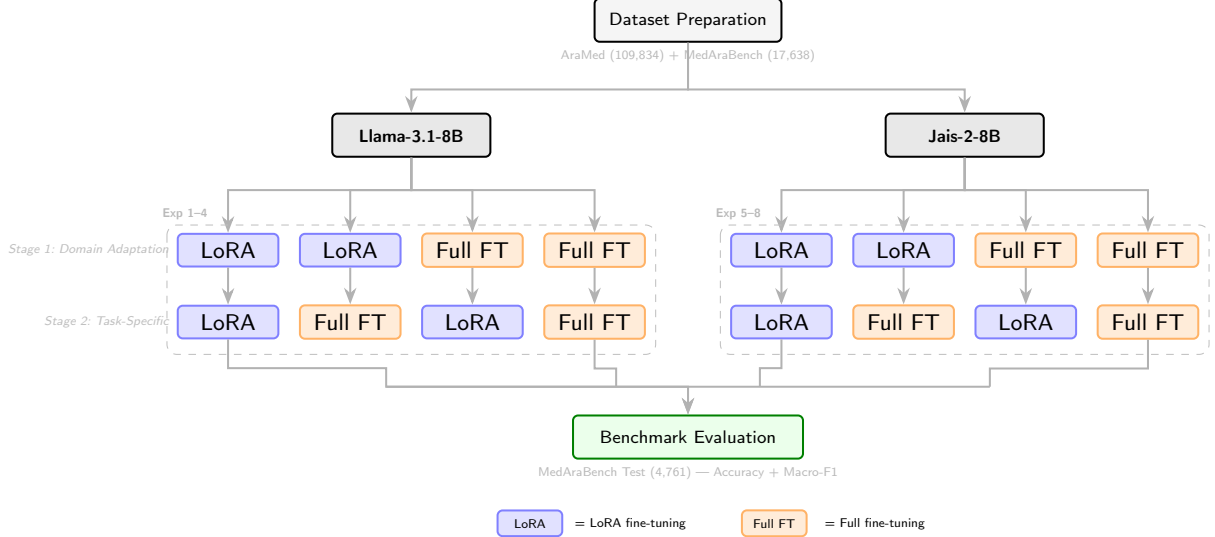


Fig. 2 Overview of the two-stage fine-tuning pipeline. Each base model is trained under four configurations combining LoRA and full fine-tuning across Stage 1 (domain adaptation on AraMed) and Stage 2 (task-specific fine-tuning on MedAraBench). All configurations are evaluated on the MedAraBench test set. The diagram shows the controlled two-stage design used for the two original 8B models (Jais-2-8B and Llama-3.1-8B); the two additional models (Qwen2.5-7B and Fanar-1-9B) are evaluated in the Stage-2-only setting.

sample that has an empty question or answer is eliminated. Given that AraMed responses are free-text, letter-based validation is useless, so no answer normalization is used. A total of 107,637 samples comprise the effective training set following a 2% seeded validation holdout.

Stage 2 involves Task Fine-tuning and Evaluation (MedAraBench). MedAraBench, a five-choice multiple-choice Arabic medical benchmark that covers a variety of clinical disciplines and is modeled after medical board exams, is used in the second stage. A total of 19,891 and 4,959 samples comprise the raw training and test splits, respectively. Given that the raw dataset contains a substantial proportion of malformed answer labels, -including multi-label combinations (e.g., A+B, C or D), out-of-range letters (F, S, V), Arabic free-text responses, and empty or null values, a five-step cleaning pipeline is applied before any training or evaluation. Answer labels are normalized using a regular expression that accepts only single uppercase letters in {A, B, C, D, E}, rejecting all multi-label and free-text entries. Samples where the correct answer is E but option E is absent are removed as contradictory, samples with empty question text are dropped, and exact duplicate questions are removed retaining only the first occurrence. After cleaning, the training set

is reduced to 17,638 samples and the test set decreases to 4,761 samples, with 2,253 and 198 samples removed, respectively. Notably, this cleaning process is applied identically to training and evaluation data. This is an important correction over previous runs that had evaluated against the uncleaned test set, producing artificially deflated Macro-F1 scores due to unanswerable reference labels. After cleaning, a 5% seeded validation holdout yields an effective Stage 2 training set of 16,756 samples and a validation set of 881 samples.

4.3 Fine-tuning Configuration

4.3.1 Model Loading and Precision

Using `AutoModelForCausalLM` with `bf16` precision and `device_map="auto"` to load both base models, HuggingFace Accelerate automatically shards the model over all available GPUs. The `bf16` format is the ideal precision for big model fine-tuning since it preserves a broader dynamic range than `float16` while reducing memory consumption by 50% compared to full `float32`. Jais-2-8B-Chat comes with special modeling code and requires the `trust_remote_code=True` flag. By default, Llama-3.1-8B does not have a dedicated pad token for tokenizer configuration; instead, the EOS token is utilized as the pad token

Table 1 Dataset summary after preprocessing. Train and validation splits are used during training; the test split is held out exclusively for evaluation. The train and validation columns sum to one sample fewer than the cleaned totals of 109,834 (AraMed) and 17,638 (MedAraBench) because the size of the seeded validation holdout is obtained by flooring the 2% and 5% fractions.

Dataset	Stage	Train	Validation	Test
AraMed	Stage 1 (Domain Adaptation)	107,637	2,196	—
MedAraBench	Stage 2 (Task Fine-tuning)	16,756	881	4,761

in accordance with the typical HuggingFace procedure. Left padding is used during evaluation to align the final real token throughout batched generation, whereas right padding is utilized during training to guarantee that the causal language modeling loss is computed accurately at the right edge of the sequence.

4.3.2 LoRA Configuration

For all LoRA conditions, adapters are applied to all seven linear projection layers in each transformer block: the attention projections (**q_proj**, **k_proj**, **v_proj**, **o_proj**) and the MLP projections (**gate_proj**, **up_proj**, **down_proj**). This broader target set follows the empirical recommendation of [6], who demonstrate that targeting all linear layers outperforms the attention-only configuration proposed in the original LoRA paper [11]. The adapter rank is set to $r = 16$ with scaling factor $\alpha = 32$, yielding a ratio of $\alpha/r = 2.0$ as recommended, and a dropout of 0.05 is applied as light regularization on the low-rank update. With these settings, LoRA trains 44,302,336 parameters on Jais-2-8B, approximately 0.54% of the 8.13B total parameters for that model, while keeping all base model weights frozen. The adapter budget is architecture-dependent rather than common to all four base models: they differ in hidden size, layer count, MLP width, and key/value head configuration, so both the absolute adapter size and its share of the total vary. Under the same rank-16, all-linear target set, Llama-3.1-8B trains 41,943,040 adapter parameters, or 0.52% of its 8.03B total, and the counts for Fanar-1-9B and Qwen2.5-7B differ correspondingly; in all cases the trainable fraction stays below 1% of the base model.

4.3.3 Hyperparameters

Table 2 presents the full hyperparameter configuration for both fine-tuning strategies. The two configurations share the same optimizer (8-bit

AdamW [5]), learning rate scheduler (cosine decay), warmup ratio (3%), sequence length limit (512 tokens), and random seed (42), but differ in several key respects driven by their memory and optimization profiles. LoRA uses a learning rate of 2×10^{-4} -ten times higher than full fine-tuning’s 2×10^{-5} -because the low-rank constraint restricts the parameter space and requires a larger step size to converge. To retain the same effective batch size of 16 for both techniques, full fine-tuning uses a smaller per-device batch size of 4 (as opposed to 8 for LoRA) with proportionately more gradient accumulation steps (4 versus 2). LoRA’s low-rank restriction offers adequate implicit regularization; a weight decay of 0.01 is only used for complete fine-tuning. The data confirm the maximum sequence length of 512 tokens. Sequences in MedAraBench MCQ are much shorter, but in AraMed, they are below 450 tokens at the 99th percentile. Continuous pretraining is performed, followed by Stage 1 domain adaptation on AraMed for 1 epoch, resulting in about 6728 optimizer steps with an effective batch size of 16. Fine-tuning of stage 2 tasks is performed on the cleaned MedAraBench training split for 3 epochs. This epoch count demonstrates a steady decrease of validation loss until the final epoch.

Table 2 Settings that differ between LoRA and full fine-tuning. All other hyperparameters are shared across both strategies (see text).

Hyperparameter	LoRA	Full Fine-tuning
Learning rate	2×10^{-4}	2×10^{-5}
Weight decay	0.0	0.01
Per-device batch size	8	4
Gradient accum. steps	2	4

4.3.4 Stage Transition Logic

How the Stage 1 checkpoint is loaded for Stage 2 training is a crucial implementation element. Depending on the mix of the Stage 1 and Stage 2 methods, there are four two-stage transition paths; the Stage-2-only baselines involve no transition, since they load the pretrained base model directly. For the LoRA $\rightarrow$ LoRA transition, the Stage 1 adapter is first merged back into the base model weights via `merge_and_unload()`, after which a fresh trainable LoRA adapter is applied to the merged base. This merge step is essential: loading a `PeftModel` checkpoint directly sets all adapter parameters to `requires_grad=False`, which would result in zero trainable parameters and a degenerate training run. For the LoRA $\rightarrow$ Full transition, the Stage 1 adapter is similarly merged before setting all parameters as trainable. For Full $\rightarrow$ LoRA, the saved full model is loaded and a fresh LoRA adapter is applied directly. For Full $\rightarrow$ Full, the saved model is loaded with all parameters set to `requires_grad=True`. In all LoRA configurations, `enable_input_require_grads()` is activated on the base model prior to adapter application; This guarantees gradient flow through the checkpointing border to the adapter parameters, which is necessary when combining gradient checkpointing with LoRA on a static foundation. Two validation checks are performed before training to ensure that the optimizer has received a non-empty parameter set and that the trainable parameter percentage is within the expected range (0.01–10% for LoRA, >99% for complete fine-tuning). If either check fails, the job is immediately terminated.

4.3.5 Prompt Templates

The pipeline uses two different prompt formats. Each AraMed inquiry and its free-text doctor response are paired at Stage 1 using an open-ended approach. A structured multiple-choice question (MCQ) format that offers every potential response is used in Stage two for training and evaluation. The answer token is not included in the evaluation process for log-probability scoring.

Option E is conditionally included only when present, as a subset of MedAraBench questions contains only four answer choices. During evaluation, the prompt is truncated at the `### Answer:` header, and the predicted answer is determined

by comparing the log-probabilities of the five candidate tokens A–E, irrespective of whether the item itself presents four or five options; Section 4.4 makes the consequences of this choice explicit.

4.3.6 Memory Optimizations

Various memory reduction strategies are needed to fit an 8B model on the hardware that is currently available. Using bfloat16 mixed-precision training, the typical model footprint is decreased to roughly 16 GB. By storing first and second moments in 8-bit precision, the 8-bit AdamW optimizer [5] decreases optimizer state memory by almost 50%. Gradient checkpointing recalculates activations during the backward pass rather than storing them, saving roughly 40% of activation memory at the cost of additional computation. A single 24 GB A10G GPU can accommodate the whole memory budget for LoRA, including base model, adapter weights, optimizer states, activations, and gradients. The training effort must be divided across all four A10G GPUs (96 GB) using `device_map="auto"` in order to achieve full fine-tuning, which involves gradients and optimizer states over all 8B parameters, bringing the total to about 48 GB without checkpointing.

4.4 Evaluation Protocol

All models are evaluated on the cleaned MedAraBench test set of 4,761 samples using a log-probability selection method rather than autoregressive text generation. For each test sample, an input-only prompt is constructed using the Stage 2 MCQ format with the `### Answer:` header present but the answer token omitted. A single forward pass is performed through the model under `torch.no_grad()`, and the logits at the final prompt position are read directly. The predicted answer is the letter whose corresponding vocabulary token achieves the highest logit among the five candidates [4]:

$$\hat{y} = \arg \max_{l \in \{A, B, C, D, E\}} \text{logits}[\text{tokenId}(l)]. \quad (1)$$

No softmax function is applied, as the argmax remains unchanged under monotonic transformations. The candidate set in Equation 1 is the same

Table 3 Prompt templates for Stage 1 (AraMed) and Stage 2 (MedAraBench).

Stage 1-AraMed (training)	Stage 2-MedAraBench (train / eval)
<pre> ### Question: {Arabic question} ### Answer: {free-text answer} </pre>	<pre> ### Question: {Arabic MCQ} ### Options: A) {option A} B) {option B} C) {option C} D) {option D} E) {option E} ### Answer: C <- omitted during eval </pre>

five letters for every test item, including the minority of items that present only four answer choices: the scoring procedure does not mask E for those items, so the model can in principle place its highest logit on a letter that was never offered. Such a prediction is necessarily counted as an error, which makes the protocol slightly conservative rather than optimistic, and it may contribute to E behaving as the weakest answer class in the per-class results of Section 5.6. Restricting the candidate set to the options actually present for each item would be a marginally more forgiving protocol; because the present one is applied identically to every model and configuration, it does not affect any of the comparisons drawn in this study. This approach requires resolving the correct vocabulary token ID for each answer letter, which varies among tokenizers. For each letter, the protocol first attempts a single-token encoding of the plain form (e.g., "A"), then the space-prefixed form (e.g., " A"), falling back to the first subword ID otherwise. For Llama-3.1, the five answer letters correspond to consecutive single-token IDs (32–36), confirming clean resolution.

Left padding is applied during evaluation, in contrast to the right padding used during training, to ensure that the last real token of each prompt is aligned at position -1 across all samples in the batch. This step guarantees that the logit slice is extracted from the correct position, regardless of variations in sequence length within the batch. Inference is performed in batches of 16 samples, with a maximum tokenization length

of 2,048 tokens. However, MedAraBench prompts remain far below this limit in practice.

Two metrics are reported for all experimental conditions. Accuracy is the fraction of test samples for which the predicted letter matches the reference label. Macro F1 is the unweighted mean of per-class F1 scores across all five answer labels [18]:

$$F1_{\text{macro}} = \frac{1}{5} \sum_{l \in \{A, B, C, D, E\}} F1_l. \quad (2)$$

Macro F1 is the primary metric because it penalizes class collapse. A model that always predicts a single letter would achieve non-trivial accuracy by chance but near-zero macro F1, because precision for the four unpredicted classes diminishes to zero. Both metrics are computed using scikit-learn with `zero_division=0` to manage any class lacking predictions smoothly. The random baseline for a five-class uniform distribution is 20.0% accuracy and approximately 20.0% macro F1.

The log-probability selection process is better than text generation for three reasons. First, it is completely deterministic; sample temperature, nucleus sampling settings, or beam search configuration do not influence the prediction; it only depends on the model weights and the input. Second, evaluation is expedited because it only needs one forward pass per sample as opposed to several autoregressive stages. Third, it requires no output parsing, removing the risk of the model generating "C. Because the mechanism involves..."

instead of a bare "C", which would require regex extraction and introduce its own failure modes. We analyze all 22 experimental conditions under the same technique to provide a fair comparison among fine-tuning strategies and base models.

5 Results

5.1 Performance Across Adaptation Strategies

Table 4 and Figure 3 present the accuracy and Macro F1 scores on the cleaned MedAraBench test set (4,761 samples) under all experimental conditions. The results are grouped into three blocks: zero-shot baselines, Stage 2 fine-tuning without domain adaptation, and the full two-stage pipeline. Macro F1 is the primary metric throughout, as it penalizes class imbalance and treats all five answer classes equally, regardless of their frequency in the test set. Beyond the two original 8B models, we also report zero-shot and Stage-2-only results for two further open-weight models in the 7–9B range (Qwen2.5-7B-Instruct and Fanar-1-9B-Instruct) and two 4-bit QLoRA conditions.

The Accuracy and Macro F1 exhibit a strong correlation but not exactly. In all fine-tuned setups, accuracy is higher than Macro F1 by about 1 to 2 percentage points (such as Jais-2-8B Full+Full accuracy at 52.9% versus Macro F1 at 51.4%), which is due to slight class imbalance in the predicted labels and not a collapse of the answer classes. Zero-shot setups have the biggest discrepancy between the two metrics, which is logical because non-fine-tuned models use only a limited set of answer letter options. When fine-tuning for a specific task, the difference between accuracy and Macro F1 is constant in all configurations, indicating that fine-tuned models produce predictions that are evenly distributed across all classes.

For the two original 8B models (Jais-2-8B and Llama-3.1-8B), the configurations that end in full fine-tuning at Stage 2 cluster tightly in the 50%–51% Macro F1 range, whereas the four two-stage configurations that end in LoRA at Stage 2 are lower, at 45.0–47.1%. All fine-tuned configurations nevertheless remain well ahead of the zero-shot baselines of 34.7% (Llama-3.1-8B) and 41.2% (Jais-2-8B). The two additional models reach substantially higher Stage-2-only scores, with Fanar-1-9B and Qwen2.5-7B attaining 55.9%

and 58.8% Macro F1, respectively, under Stage-2-only LoRA. Within the two-stage factorial design applied to these two original models, the best configuration is Jais-2-8B Full+Full at 51.4% Macro F1, followed closely by Llama-3.1-8B None+Full at 51.0%, with a margin of less than a point separating the strongest configurations despite substantial differences in adaptation strategies. This is not the highest score we obtain anywhere: both additional model families are stronger under Stage-2-only adaptation, and a single change to the Jais-2-8B Stage-2 LoRA configuration (dropout set to zero) reaches 52.3% Macro F1 (Section 5.4, Table 6). Jais-2-8B Full+Full is therefore best described as the strongest point of the factorial design rather than the strongest result overall. Figure 4 visualizes these results, clearly delineating the disparity between fine-tuned and zero-shot conditions and showing how the two models converge with the implementation of task-specific fine-tuning.

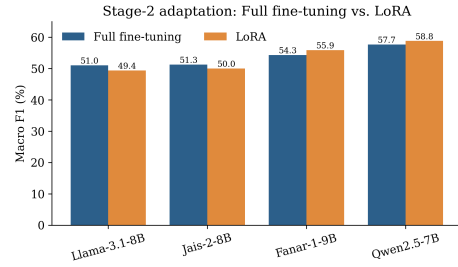


Fig. 3 Stage-2 adaptation across all four base models: full fine-tuning versus LoRA (Macro F1, %). The full-fine-tuning advantage is confined to the original Jais and Llama models; for Fanar-1-9B and Qwen2.5-7B even a default-configuration LoRA matches or exceeds full fine-tuning.

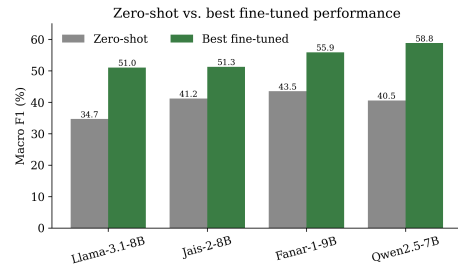


Fig. 4 Zero-shot versus best fine-tuned Macro F1 (%) per base model. Task fine-tuning erases the zero-shot advantage of the Arabic-centric models, with the general-purpose Qwen2.5-7B attaining the strongest result after adaptation.

Table 4 Results on the MedAraBench test set (4,761 samples), reported as Accuracy and Macro F1 (%). S1 and S2 denote the Stage 1 and Stage 2 adaptation methods (None, LoRA, QLoRA, or Full). Zero-shot rows apply no fine-tuning. The two additional models (Qwen2.5-7B, Fanar-1-9B) are evaluated only in the Stage-2-only setting. Bold marks the best two-stage configuration for each original 8B model. [†] marks a Full-versus-LoRA contrast that remains reliable under three-seed reruns (Table 5) and paired significance testing; the corresponding Jais-2-8B contrast falls within seed-level variation.

Model	S1	S2	Acc. (%)	F1 (%)
<i>Zero-shot (no fine-tuning)</i>				
Llama-3.1-8B	—	—	37.2	34.7
Qwen2.5-7B	—	—	42.5	40.5
Jais-2-8B	—	—	42.4	41.2
Fanar-1-9B	—	—	45.4	43.5
<i>Stage 2 only (no domain adaptation)</i>				
Llama-3.1-8B	None	LoRA	50.3	49.4
Llama-3.1-8B	None	QLoRA	52.3	51.3
Llama-3.1-8B	None	Full	52.0	51.0 [†]
Qwen2.5-7B	None	LoRA	59.9	58.8
Qwen2.5-7B	None	Full	58.3	57.7
Jais-2-8B	None	LoRA	51.1	50.0
Jais-2-8B	None	QLoRA	53.1	51.3
Jais-2-8B	None	Full	53.2	51.3
Fanar-1-9B	None	LoRA	57.3	55.9
Fanar-1-9B	None	Full	55.8	54.3
<i>Two-stage pipeline (original 8B models)</i>				
Llama-3.1-8B	LoRA	LoRA	46.2	45.5
Llama-3.1-8B	LoRA	Full	51.3	50.3
Llama-3.1-8B	Full	LoRA	45.6	45.0
Llama-3.1-8B	Full	Full	51.6	50.5
Jais-2-8B	LoRA	LoRA	48.2	47.1
Jais-2-8B	LoRA	Full	52.1	50.0
Jais-2-8B	Full	LoRA	47.3	46.8
Jais-2-8B	Full	Full	52.9	51.4

5.2 Per-Specialty Analysis

To understand the advantages of full fine-tuning compared with LoRA, we analyze Stage 2 performance at the finer of the two levels of the MedAraBench taxonomy described in Section 2.2, namely its 26 fine-grained specialty labels, excluding one specialty with too few test samples for a reliable estimate and leaving 25. Figure 5 reports, for each specialty, the difference in per-specialty accuracy between full fine-tuning and LoRA at Stage 2, using the Stage-2-only configurations to determine the effect of the adaptation method.

The advantage of full fine-tuning is generally positive for both base models (exhibiting a mean difference of 3.1 points for Llama-3.1-8B and 2.9 points for Jais-2-8B) and aligns in a clear majority

of specialties (16 of 25 for Llama-3.1-8B and 18 of 25 for Jais-2-8B). The largest improvements are shown in clinically dense, reasoning-heavy specialties such as Emergency Medicine and Oncology, although they are not uniform across the two models: in Emergency Medicine both models gain more than 20 points from full fine-tuning, whereas in Oncology the gain is large for Llama-3.1-8B (about 18 points) but considerably smaller for Jais-2-8B (about 9 points). This is consistent with the distribution-shift account from Section 3: the specialties that demand the most adaptation away from the base model’s pretraining distribution are precisely those where the extensive full-parameter update budget is most advantageous, and where the rank-16 LoRA constraint is most limiting.

Simultaneously, the per-specialty gaps are noisy. A minority of specialties show a negative gap for one or both models, and the magnitude of individual specialty differences varies widely. Given the limited number of test questions in several specialties, individual per-specialty estimates should be considered suggestive rather than definitive, and we avoid making robust conclusions from any single discipline. The prevailing trend indicates that, across all specialties, full fine-tuning is the most reliable choice, applicable to the Arabic-focused and the English-dominant base models, reinforcing the primary conclusion that the adaptation method is more significant than pretraining language at this scale.

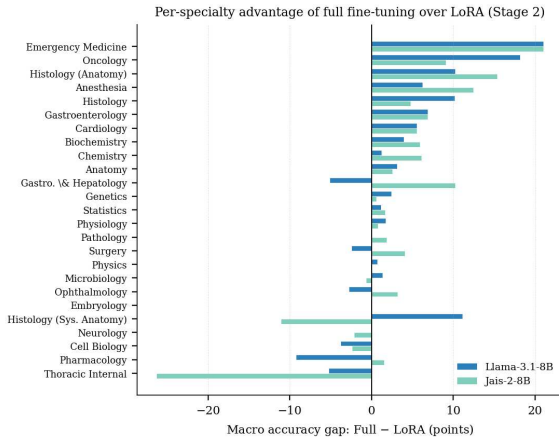


Fig. 5 Per-specialty advantage of full fine-tuning over LoRA at Stage 2 (Stage-2-only configurations, no domain adaptation). Bars show the difference in per-specialty accuracy (Full – LoRA) in points; positive values indicate that full fine-tuning outperforms LoRA. Specialties are ordered by the mean gap across the two models.

5.3 Reliability of the Full-versus-LoRA Gap

Given that the differences between full fine-tuning and LoRA are numerically small, we assess how much of the gap is stable rather than an artifact of a single training run. We rerun the four Stage-2-only configurations for the original 8B models with three random seeds (42, 1337, and 2024) and summarize the outcome in Table 5 and Figure 6. For Llama-3.1-8B, full fine-tuning stays ahead of LoRA by about 1.5 points of Macro F1

(51.3 versus 49.7 on average), a margin several times larger than the seed-level standard deviation, so this contrast is reliable and marked with a dagger in Table 4. For Jais-2-8B, the picture is weaker: the mean gap is around 1.3 points, but the LoRA condition alone varies by about one point across seeds, so the two conditions overlap within a single standard deviation, and we do not treat the Jais contrast as established. Applying a paired significance test (McNemar’s test [15]) and 95% bootstrap confidence intervals [7] to the test predictions leads to the same qualitative conclusion. Therefore, the headline advantage of full fine-tuning is best described as small and model-dependent rather than uniform.

Table 5 Reliability of the Stage-2-only Full-versus-LoRA comparison under three random seeds (42, 1337, 2024). Values are mean $\pm$ standard deviation across seeds. For Llama-3.1-8B the Full advantage exceeds seed-level variation for both metrics, whereas for Jais-2-8B the two conditions overlap within one standard deviation.

Model	S2	Acc. (%)	F1 (%)
Llama-3.1-8B	LoRA	50.4 $\pm$ 0.26	49.7 $\pm$ 0.30
Llama-3.1-8B	Full	52.2 $\pm$ 0.28	51.3 $\pm$ 0.34
Jais-2-8B	LoRA	51.6 $\pm$ 0.71	50.5 $\pm$ 1.02
Jais-2-8B	Full	53.4 $\pm$ 0.22	51.8 $\pm$ 0.57

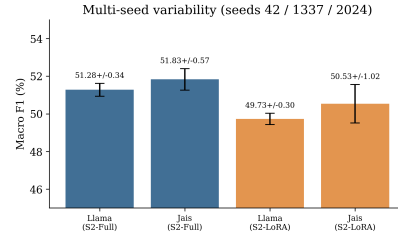


Fig. 6 Macro F1 (%) for the four Stage-2-only configurations rerun with three random seeds (42, 1337, 2024); bars show the mean and error bars the standard deviation. The Full advantage exceeds seed-level variation for Llama-3.1-8B but overlaps within one standard deviation for Jais-2-8B.

5.4 Is LoRA Under-tuned?

A single default LoRA configuration cannot, on its own, establish that LoRA is inherently weaker than full fine-tuning. To separate the method from

its hyperparameters, we sweep the rank, learning rate, dropout, and target modules of the Stage-2 LoRA adapter for both original models (Table 6). Two factors clearly matter. Increasing the rank from the default 16 to 64 raises Macro F1 to 51.3% for Llama-3.1-8B and 51.5% for Jais-2-8B (Figure 7), and setting dropout to zero raises Jais-2-8B to 52.3% Macro F1, which slightly exceeds its full fine-tuning result. An aggressive learning rate of 4×10^{-4} reduces performance for both models, and restricting the adapter to attention projections only is consistently worse than adapting all linear layers. In summary, much of the disparity observed at the default setting reflects an under-tuned adapter rather than a definitive limitation of low-rank adaptation, which further justifies the conditional assertion of the full-versus-LoRA recommendation.

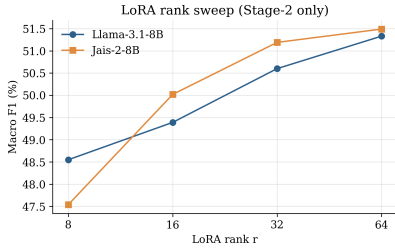


Fig. 7 Stage-2-only LoRA Macro F1 (%) as a function of adapter rank r for the two original 8B models. Raising the rank from the default 16 to 64 recovers most of the gap to full fine-tuning, indicating that the default LoRA configuration is under-tuned rather than fundamentally weaker.

We also test quantized adaptation by training 4-bit QLoRA adapters for both original models (the QLoRA rows in Table 4). QLoRA reaches 51.3% Macro F1 for Llama-3.1-8B and 51.3% for Jais-2-8B, matching full fine-tuning while using far less memory. This indicates that the gap at the default LoRA setting is not attributable to quantization, and a memory-efficient adapter can be competitive once it is adequately configured.

5.5 Diagnosing the Stage 1 Null Result

Stage 1 continued pre-training on AraMed yields no reliable downstream benefit. Table 7 shows that Stage 1 does adapt the base model as intended: AraMed held-out perplexity falls sharply, from 11.4 to 5.0 for Llama-3.1-8B and from 20.1 to

9.6 for Jais-2-8B under full Stage 1 fine-tuning. Despite this, downstream Macro F1 after full Stage 2 fine-tuning does not improve over direct Stage-2-only fine-tuning for either model, and conditioning on a LoRA Stage 1 checkpoint is slightly worse. Extending Stage 1 from one to three epochs for Jais-2-8B does not help either (51.0% versus 51.3% for direct Stage 2). The most consistent explanation is a task-format mismatch: AraMed consists of open-ended doctor-patient consultation text, whereas MedAraBench is five-way MCQ exam data evaluated by log-probability, so fluency gained on free-form Arabic medical prose does not transfer to structured answer selection. Additional epochs mainly risk eroding the base model’s instruction-following behavior rather than adding transferable exam knowledge.

5.6 Error Analysis

Beyond the per-specialty view of Section 5.2, we report a finer-grained error analysis for the best-performing configuration.

Two distinct patterns emerge. Class D is the strongest (57.2% F1), whereas class E is the weakest (43.0% F1); this result is consistent with E being the rarest label, as a subset of MedAraBench questions offers only four options. Off-diagonal errors are distributed fairly uniformly among the distractor classes instead of being concentrated in any single confusion pair, indicating that the omissions reflect genuine knowledge gaps rather than a positional or answer-format bias. Together with the year-level breakdown below, the pronounced drop on the Y3–Y5 questions (Table 10) indicates that the remaining errors are concentrated in the more advanced clinical-reasoning content rather than in Y1 factual recall.

We further analyze accuracy by the academic-year difficulty label (Y1–Y5) annotated in MedAraBench, which is important because the benchmark is heavily skewed toward the introductory Y1 level (roughly 60% of the test questions). Table 10 reports annual accuracy for the optimal configuration (Jais-2-8B, full+full). Performance peaks on the abundant Y1 and Y2 questions (55.6% and 56.4%) and declines on the less frequent, more advanced years (37.0% on Y3, 46.5% on Y4, and 41.3% on Y5), indicating that the overall accuracy is supported by the Y1-dominated distribution and

Table 6 LoRA hyperparameter sweep at Stage 2 (no domain adaptation) for the two original 8B models. The baseline configuration is rank 16, learning rate 2×10^{-4} , dropout 0.05, and all linear projections as target modules; each subsequent row changes a single factor. For reference, the corresponding full fine-tuning Macro F1 is 51.0% for Llama-3.1-8B and 51.3% for Jais-2-8B. Higher rank and lower dropout recover most of the gap, while an excessive learning rate (4×10^{-4}) and attention-only targeting degrade performance.

Configuration	Llama-3.1-8B		Jais-2-8B	
	Acc.	F1	Acc.	F1
Baseline (r16, lr2e-4, do0.05)	50.3	49.4	51.1	50.0
Rank 8	49.1	48.5	49.1	47.5
Rank 32	52.0	50.6	52.8	51.2
Rank 64	52.8	51.3	52.8	51.5
Learning rate 1e-4	51.9	50.5	52.5	51.1
Learning rate 4e-4	45.4	44.5	42.3	42.0
Dropout 0.0	52.2	50.5	54.2	52.3
Dropout 0.1	52.0	51.0	53.3	51.4
Attention-only targets	47.3	46.1	50.8	49.5

Table 7 Stage 1 diagnostics. Panel (a): AraMed held-out perplexity before and after Stage 1 continued pre-training, showing that Stage 1 substantially adapts the language model. Panel (b): downstream Macro F1 (%) after full Stage 2 fine-tuning as a function of the Stage 1 checkpoint, including a three-epoch Stage 1 variant for Jais-2-8B. Despite the large perplexity reduction, no Stage 1 checkpoint improves downstream MCQ performance over direct Stage-2-only fine-tuning (None).

<i>(a) AraMed perplexity</i>		
Model	Before	After (Full / LoRA)
Llama-3.1-8B	11.4	5.0 / 5.2
Jais-2-8B	20.1	9.6 / 9.8

<i>(b) Stage-1 checkpoint $\rightarrow$ Stage-2-Full Macro F1</i>		
Stage 1	Llama-3.1-8B	Jais-2-8B
None (direct S2)	51.0	51.3
LoRA	50.3	50.0
Full (1 epoch)	50.5	51.4
Full (3 epochs)	—	51.0

Table 8 Per-class precision, recall, and F1 (%) for the best configuration (Jais-2-8B, full+full) on the MedAraBench test set.

Answer class	Precision (%)	Recall (%)	F1 (%)
A	54.6	53.7	54.1
B	50.1	54.4	52.2
C	50.2	50.4	50.3
D	58.9	55.6	57.2
E	45.1	41.1	43.0

that the model exhibits significant deficiencies in higher-year clinical reasoning.

To characterize what kinds of errors produced by the optimal configuration, we manually coded a sample of its misclassifications. Starting with the

2,241 misclassified items (predicted $\neq$ gold) of the Jais-2-8B Full+Full run, we drew a random sample of 100 errors (seed 42), stratified by specialty and academic year and deliberately over-sampling class E and the Y3–Y5 levels so that the sample covers

Table 9 Confusion matrix (row: true class, column: predicted class) for the best configuration (Jais-2-8B, full+full) on the MedAraBench test set.

	Pred A	Pred B	Pred C	Pred D	Pred E
True A	614	201	166	138	25
True B	169	622	192	134	26
True C	173	218	569	139	30
True D	129	169	169	623	31
True E	40	31	38	23	92

Table 10 Accuracy (%) by academic-year difficulty level (Y1–Y5) for the best configuration (Jais-2-8B, full+full) on the MedAraBench test set.

	Y1	Y2	Y3	Y4	Y5
Accuracy (%)	55.6	56.4	37.0	46.5	41.3

the weakest label (Table 8) and the harder years (Table 10) rather than being predominantly influenced by abundant Y1 items. Each sampled error was read in Arabic, including the question, options, gold answer, and model prediction, and assigned to exactly one of four categories: language understanding, medical knowledge, clinical reasoning, or answer-format/annotation. Of the 100 errors, 80 were attributed to deficiencies in medical knowledge, 11 were due to failures in clinical reasoning, 6 were related to answer-format/annotation artifacts, and 3 were caused by language understanding issues. In 7 items, we assessed the gold label itself to be wrong or genuinely ambiguous (the annotation-error subcount, which included the format and language categories).

Table 11 Manual coding of a stratified random sample of 100 Jais-2-8B Full+Full errors (seed 42), by category, with the annotation-error subcount.

Error category	Count (of 100)
Medical knowledge	80
Clinical reasoning	11
Answer-format / annotation	6
Language understanding	3
<i>of which: gold label wrong/ambiguous</i>	7

The coding connects directly to the quantitative tables. Errors in medical knowledge dominate (80%), consistent with the Stage 1 null

result (Section 5.5, Table 7): significant improvements resulted from task-format supervision on MedAraBench, whereas open-ended AraMed prose did not supply the structured exam information the model lacked. The failures in clinical reasoning are concentrated in the harder years; specifically, 9 of the 11 occur in Y3–Y5, reflecting the decline in accuracy at those levels in Table 10, indicating that the complex multi-step case-based questions, rather than the single-fact Y1 items, are where reasoning breaks down. Finally, the non-trivial annotation-error subcount (7 of 100) is a direct symptom of the data-quality issues associated with MedAraBench already noted in Section 2.2, the benchmark was assembled by manually digitizing scanned examinations, and a fraction of items are mislabeled, distorted, or dependent on context.

Representative examples illustrate each category. Medical knowledge: a pharmacology item on the intravenous dose equivalent of 200 mg of oral propranolol; the required conversion factor is a specific detail the model did not know, so it selected a plausible but wrong value. Clinical reasoning: a surgical case study detailing a patient with asthma on repeated steroid courses who develops hirsutism, truncal obesity, and easy bruising; recognizing this as (iatrogenic) Cushing’s syndrome requires linking the symptoms, yet the model instead chose Conn’s syndrome. Language understanding: a microbiological item whose stem contained a distorted transliteration of *Cryptosporidium parvum*, concealing the organism in question. Answer-format / annotation: a physiology item asking what increases after a large water intake whose gold key was ADH, which is physiologically backwards (ADH falls with a water load), so the item is inaccurately labeled.

5.7 Analysis and Discussion

Fine-tuning consistently improves zero-shot performance. Fine-tuning on MedAraBench considerably improves both models independent of the method. Llama-3.1-8B improves from 34.7% to a peak of 51.3% Macro F1 (+16.6 points; its 4-bit QLoRA configuration, versus 51.0% under full fine-tuning), and Jais-2-8B improves from 41.2% to 51.4% (+10.2 points). As a result of its Arabic-native pretraining, the elevated zero-shot baseline of Jais-2-8B can be predicted, but the gap is significantly reduced after fine-tuning, indicating that task-specific adaptation is the major factor at this scale.

Stage 1 does not reliably achieve domain adaptation. Although we believe that the AraMed pretraining provides a useful inductive bias for downstream medical QA, the two-stage pipeline does not outperform direct Stage 2 fine-tuning in most scenarios. For Llama-3.1-8B, the best two-stage outcome (Full+Full: 50.5% F1) is somewhat worse than the Stage 2 only full fine-tuning baseline (51.0% F1). For Jais-2-8B, the best two-stage result (Full+Full: 51.4% F1) is within 0.1 points of the Stage 2 only full baseline (51.3% F1). This pattern implies that one epoch of open-ended Arabic medical QA pretraining on the AraMed dataset is insufficient to equip the model with the ordered clinical knowledge that can be transferred to multiple-choice reasoning. In certain situations, catastrophic forgetting may even slightly impair the base model’s ability to follow instructions.

LoRA is weaker than full fine-tuning at Stage 2 in the default configuration in both models. In every paired comparison where the Stage 2 method varies while the Stage 1 method remains unchanged, full fine-tuning at Stage 2 outperforms LoRA. Llama-3.1-8B without Stage 1 produces 51.0% F1 versus 49.4% for LoRA (+1.6 points). For Jais-2-8B without Stage 1, the gap is 51.3% versus 50.0% (+1.3 points). The same behavior is observed in the two-stage pipeline, with LoRA+Full consistently outperforming LoRA+LoRA in both models. Full+Full beats Full+LoRA consistently for both models. This shows that the small parameter space of LoRA, while sufficient for many English NLP tasks, may restrict its adaptability for Arabic medical MCQs; this domain is out-of-distribution in language and task format for both base models. This

Full-versus-LoRA advantage is, however, model-dependent and configuration-dependent: as shown in Sections 5.3, 5.4, and 5.5, the gap is reliable only for one of the two original models, largely closes once the LoRA rank and dropout are tuned or 4-bit QLoRA is used, and does not replicate on Fanar-1-9B or Qwen2.5-7B, so we do not interpret it as an unqualified preference for full fine-tuning over LoRA.

LoRA at Stage 1 is worse than full fine-tuning at Stage 1 only when Stage 2 is also full fine-tuning. In that case, conditioning on a LoRA Stage 1 checkpoint results in poorer final Macro F1 scores than conditioning on a complete fine-tuning Stage 1 checkpoint, and this holds for both models: for Jais-2-8B, Full+Full (51.4%) is 1.4 points better than LoRA+Full (50.0%); for Llama-3.1-8B, Full+Full (50.5%) is 0.2 points better than LoRA+Full (50.3%). When Stage 2 is LoRA, however, the ordering reverses: LoRA+LoRA is marginally ahead of Full+LoRA for both models (Llama-3.1-8B, 45.5% versus 45.0%; Jais-2-8B, 47.1% versus 46.8%). These reversals amount to only 0.3–0.5 points, well inside the seed-level variation documented for the Stage-2-only runs in Section 5.3, so we read the pair of contrasts as evidence that the Stage 1 method has no consistent main effect rather than as a genuine advantage for LoRA at Stage 1: the Stage 1 choice matters only in interaction with the Stage 2 method. Within the full Stage 2 conditions, the quality of the Stage 1 checkpoint is more important for Jais-2-8B than for Llama-3.1-8B, presumably because Jais already has strong Arabic priors, which can be effectively maintained and expanded by full adaptation at Stage 1.

Arabic-focused pre-training yields a remarkable improvement in zero-shot performance that almost dissipates upon task-specific fine-tuning. At the zero-shot level, Jais-2-8B achieves a 41.2% Macro F1 as opposed to Llama-3.1-8B at 34.7%, where the former outperforms the latter by 6.5 due to its Arabic-centric pre-training. However, when task-specific fine-tuning with MedAraBench is considered, the difference disappears, as Jais-2-8B Full+Full reaches 51.4% Macro F1 under optimal full fine-tuning settings, while Llama-3.1-8B None+Full achieves 51.0%, resulting in a negligible 0.4 point difference. Even after accounting for their optimal two-stage fine-tuning setups, specifically

Jais-2-8B Full+Full achieving 51.4% and Llama-3.1-8B Full+Full attaining 50.5%, the difference is only 0.9 points.

Do the two headline patterns replicate across model families? The two additional models provide a mixed answer. The first pattern, where full fine-tuning outperforms LoRA, does not replicate on any of the new families for which the comparison is available: a Stage-2-only LoRA slightly outperforms full fine-tuning on Qwen2.5-7B (58.8% versus 57.7% Macro F1) and on Fanar-1-9B (55.9% versus 54.3%; Figure 3). This reinforces the reading from the seed reruns and the LoRA sweep that the full-fine-tuning advantage is specific to the original Jais and Llama setting rather than a general property. The second pattern, the erosion of the zero-shot Arabic advantage after fine-tuning, does replicate directionally (Figure 4): the Arabic-centric Fanar-1-9B starts from higher zero-shot Macro F1 (43.5%) than the general-purpose Qwen2.5-7B (40.5%). However, after task fine-tuning, the general-purpose Qwen2.5-7B attains the strongest result overall (58.8% Macro F1), indicating that task-specific adaptation, rather than the amount of Arabic pre-training, is the primary factor influencing final performance at this scale.

6 Conclusion

This study set out to give a team deploying an Arabic clinical question answering assistant a defensible answer to a budget question: how should an open-weight model be adapted when GPU capacity, rather than ideas, is the binding constraint? To answer it we conduct a systematic comparison between LoRA and full fine-tuning for Arabic medical question answering, evaluating both adaptation strategies over two stages: domain adaptation on AraMed and task-specific fine-tuning on MedAraBench. We use an English-dominant and an Arabic-centric base model, each with eight billion parameters as the controlled foundation, complemented by two additional models with 7–9 billion parameters (Fanar-1-9B and Qwen2.5-7B). Three findings emerge, with the first being more qualified than we initially expected. First, for the two original 8B models (Jais-2-8B and Llama-3.1-8B), a default-configuration full fine-tuning outperforms a default-configuration LoRA at the task-adaptation phase by 1.3–1.6

Macro F1 points; however, this gap is small compared with seed-to-seed variation (it is reliable for Llama but within seed noise for Jais), it significantly diminishes once the LoRA rank and dropout are tuned or 4-bit QLoRA is used, and it does not replicate on the two additional model families we test, where even a default-configuration LoRA matches or exceeds full fine-tuning. Second, one epoch of domain adaptation on AraMed does not yield any reliable improvement over direct task fine-tuning, and it slightly degrades performance in some settings. This suggests that open-ended Arabic medical pre-training does not transfer well to multiple-choice clinical reasoning at this scale. Third, contrary to our main hypothesis, the relative ranking of adaptation strategies remains largely unchanged regardless of the pretraining language of the base model: although Arabic-centric pretraining provides Jais-2-8B with a significant zero-shot advantage, this benefit is almost completely lost after task-specific fine-tuning. For the two original 8B models under the default adapter configuration, full fine-tuning remains the superior choice for the Arabic-native and the English-dominant models, whereas a tuned LoRA or 4-bit QLoRA proves to be competitive.

These findings indicate that, within the tested context, the choice of adaptation method is more sensitive to target-task difficulty and to the specific model than to the amount of Arabic pre-training of the base model. For a team deploying an Arabic clinical question answering assistant under a limited GPU budget, the operational recommendation that follows is therefore concrete: select the strongest available base model for the target task, adapt it with a tuned low-rank adapter or with 4-bit QLoRA on a single 24 GB GPU, and spend the remaining budget on task-specific supervision rather than on a full-parameter update or on an additional domain-adaptation stage. Full fine-tuning across four GPUs remains a reasonable default only in the narrow case of the two original 8B models adapted with an unadjusted adapter, and even there the preference is modest and context-dependent rather than a general rule, because a tuned LoRA or 4-bit QLoRA is competitive and the advantage does not extend to the other model families we evaluated. Several limitations restrict this interpretation and should be

read alongside the results. First, the study covers a single scale band (7–9 billion parameters), so we cannot say whether the same pattern holds at larger scales such as 70 billion. Second, evaluation is restricted to multiple-choice questions from AraMed and MedAraBench and does not measure open-ended clinical generation. Third, answers are selected by comparing token log-probabilities rather than by free-form generation, so the results reflect answer-selection ability rather than the quality of the underlying reasoning. Finally, the differences between the best-performing configurations are small and, in certain comparisons, fall within the margin of seed-level variation.

A further limitation concerns scope rather than experimental design, and it bears directly on the phrase clinical decision support in our title. What we measure is answer selection over curated multiple-choice examination items, not the safety of free-form clinical language: a model that assigns the highest log-probability to the correct option may still produce incomplete, misleading, or fabricated text when a clinician or a patient asks it an open question. The absolute performance levels reinforce the point, since the adapted configurations reported here reach 51–59% Macro F1 and are unevenly competent across specialties, which is far from the reliability that triage or point-of-care use would demand. Nothing in this study should therefore be read as evidence that any of these models is ready for clinician-facing or patient-facing deployment. Moving from the artefacts evaluated here to a decision-support tool would require, at a minimum, evaluation on open-ended and multi-turn clinical tasks, prospective validation in which a qualified clinician retains responsibility for every recommendation, calibrated uncertainty estimates together with an explicit abstention and escalation path when confidence is low, monitoring for distribution shift across specialties and dialects, and compliance with the medical-device and data-protection regimes of the jurisdiction of deployment, since software of this kind is regulated as a medical device in many of them. We therefore present this work as an adaptation-strategy study for one component of such a system rather than as a clinical validation of it.

The research opens several avenues for future development. First, replicating the factorial experiment conducted within the present study on an

expanded 70B parameter level, using Llama-3.1-70B and Jais-2-70B, which are publicly available, will reveal whether the apparent convergence of the two types of models after the fine-tuning step is characteristic of 8B-capacity models or extends to models with larger parameters. In addition to the scalability question, the domain adaptation phase can be investigated in great detail. The marginal or even negative effect of AraMed pretraining reported in the present experiment is related to the inconsistency between the open-ended consultation form of the AraMed dataset and the multiple choice task. Future work can focus on exploring corpora for pretraining, enhancing the representation of clinical tasks, or experimenting with different pretraining techniques, including multi-epoch or curriculum pretraining. The evaluation itself, limited by the clinical multiple choice task, can be extended to other types of Arabic NLP tasks, such as open-ended clinical question generation, dialectal Arabic, and multi-turn medical consultation dialogue. Finally, the potential influence of the chat versions of both base models used in the present work, which were fine-tuned with guidelines on instruction alignment, warrants investigation.

Acknowledgements. Our sincere appreciation goes to RISE at the University of Sharjah for their generous support throughout the course of this research. We also extend our gratitude to the MLALP Research Group for their invaluable contribution by providing the necessary workstation resources for the development of our models.

Declarations

Funding. This work was supported by RISE at the University of Sharjah.

Competing interests. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethics approval and consent to participate. Not applicable. This study uses only publicly available benchmark datasets and does not involve human participants or animals.

Consent for publication. Not applicable.

Data availability. Both datasets used in this study are publicly available from their original authors: AraMed and MedAraBench.

Code availability. All code written for this work is publicly available at <https://github.com/omaratef3221/arabic-medical-qa-research>, comprising the MedAraBench cleaning and deduplication pipeline, the two-stage domain- and task-adaptation fine-tuning scripts, and the log-probability evaluation code, together with the per-configuration prediction files required to reproduce every table and figure in this paper.

Author contributions. All authors contributed to the study conception and design. Material preparation, the experimental pipeline, and analysis were performed by Ibrahim Abaker Hashem, Omar Elgendy, Ali Bou Nassif, Ismail Shahin, Ashraf Elnagar, Ayad Turky, and Imad Afyouni. All authors contributed to the writing of the manuscript, commented on previous versions, and read and approved the final manuscript.

References

- [1] Abu-Daoud M, Kharouf L, Hajj OE, et al (2026) Medarabench: Large-scale arabic medical question answering dataset and benchmark. arXiv preprint arXiv:260201714 <https://doi.org/10.48550/arXiv.2602.01714>
- [2] Alasmari A, Alhumoud S, Alshammari W (2024) Aramed: Arabic medical question answering using pretrained transformer language models. In: Proceedings of the 6th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT) with Shared Tasks on Arabic LLMs Hallucination and Dialect to MSA Machine Translation@ LREC-COLING 2024, pp 50–56, <https://doi.org/10.63317/2k278czipbxt>
- [3] Antoun W, Baly F, Hajj HM (2020) Aragpt2: Pre-trained transformer for arabic language generation. CoRR abs/2012.15520. <https://doi.org/10.48550/arXiv.2012.15520>, URL <https://arxiv.org/abs/2012.15520>, 2012.15520
- [4] Brown T, Mann B, Ryder N, et al (2020) Language models are few-shot learners. Advances in neural information processing systems 33:1877–1901
- [5] Dettmers T, Lewis M, Shleifer S, et al (2022) 8-bit optimizers via block-wise quantization. In: International Conference on Learning Representations (ICLR)
- [6] Dettmers T, Pagnoni A, Holtzman A, et al (2023) Qlora: Efficient finetuning of quantized llms. Advances in neural information processing systems 36:10088–10115
- [7] DiCiccio TJ, Efron B (1996) Bootstrap confidence intervals. Statistical Science 11(3):189–228. <https://doi.org/10.1214/ss/1032280214>
- [8] Elgendy O, Nassif AB (2023) Alzheimer detection using different deep learning methods with mri images. In: 2023 Advances in Science and Engineering Technology International Conferences (ASET), IEEE, pp 1–6, <https://doi.org/10.1109/ASET56582.2023.10180640>
- [9] Grattafiori A, Dubey A, Jauhri A, et al (2024) The llama 3 herd of models. arXiv preprint arXiv:240721783 <https://doi.org/10.48550/arXiv.2407.21783>
- [10] Han T, Adams LC, Papaioannou JM, et al (2025) Medalpaca – an open-source collection of medical conversational ai models and training data. <https://doi.org/10.48550/arXiv.2304.08247>, URL <https://arxiv.org/abs/2304.08247>, arXiv:2304.08247
- [11] Hu EJ, Shen Y, Wallis P, et al (2022) LoRA: Low-rank adaptation of large language models. In: International Conference on Learning Representations (ICLR)
- [12] Jin D, Pan E, Oufattole N, et al (2021) What disease does this patient have? a large-scale open domain question answering dataset from medical exams. Applied Sciences 11(14):6421. <https://doi.org/10.3390/app11146421>
- [13] Jin Q, Dhingra B, Liu Z, et al (2019) Pubmedqa: A dataset for biomedical research question answering. <https://doi.org/10.48550/arXiv.1909.06146>, URL <https://arxiv.org/abs/1909.06146>, arXiv:1909.06146

- [14] Luo R, Sun L, Xia Y, et al (2022) Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics* 23(6). <https://doi.org/10.1093/bib/bbac409>, URL <http://dx.doi.org/10.1093/bib/bbac409>
- [15] McNemar Q (1947) Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* 12(2):153–157. <https://doi.org/10.1007/BF02295996>
- [16] Nassif AB, Talib MA, Nasir Q, et al (2022) Breast cancer detection using artificial intelligence techniques: A systematic literature review. *Artificial intelligence in medicine* 127:102276. <https://doi.org/10.1016/j.artmed.2022.102276>
- [17] Pal A, Umapathi LK, Sankarasubbu M (2022) Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In: Flores G, Chen GH, Pollard T, et al (eds) *Proceedings of the Conference on Health, Inference, and Learning, Proceedings of Machine Learning Research*, vol 174. PMLR, pp 248–260, URL <https://proceedings.mlr.press/v174/pal22a.html>
- [18] Pedregosa F, Varoquaux G, Gramfort A, et al (2011) Scikit-learn: Machine learning in python. *the Journal of machine Learning research* 12:2825–2830
- [19] Qwen, Yang A, Yang B, et al (2024) Qwen2.5 technical report. *arXiv preprint arXiv:241215115* <https://doi.org/10.48550/arXiv.2412.15115>
- [20] Sengupta N, Sahu SK, Jia B, et al (2023) Jais and jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models. *arXiv preprint arXiv:230816149* <https://doi.org/10.48550/arXiv.2308.16149>
- [21] Shuttleworth R, Andreas J, Torralba A, et al (2024) Lora vs full fine-tuning: An illusion of equivalence. *arXiv preprint arXiv:241021228* <https://doi.org/10.48550/arXiv.2410.21228>
- [22] Singhal K, Tu T, Gottweis J, et al (2025) Toward expert-level medical question answering with large language models. *Nature medicine* 31(3):943–950. <https://doi.org/10.1038/s41591-024-03423-7>
- [23] Team F, Abbas U, Ahmad MS, et al (2025) Fanar: An arabic-centric multimodal generative ai platform. *arXiv preprint arXiv:250113944* <https://doi.org/10.48550/arXiv.2501.13944>
- [24] Torfa W, Elgendy O, Nassif AB (2025) Enhancing rlhf in llms: Comparing bert, xgboost, and deep models for bias detection. In: *2025 18th International Conference on Development in eSystem Engineering (DeSE)*, pp 226–231, <https://doi.org/10.1109/DeSE68208.2025.11367955>
- [25] Wu C, Lin W, Zhang X, et al (2023) Pmc-llama: Towards building open-source language models for medicine. <https://doi.org/10.48550/arXiv.2304.14454>, URL <https://arxiv.org/abs/2304.14454>, *arXiv:2304.14454*
- [26] Yao Z, Zhang Z, Tang C, et al (2024) Medqacs: Objective structured clinical examination (osce)-style benchmark for evaluating llm clinical skills. <https://doi.org/10.48550/arXiv.2410.01553>, URL <https://arxiv.org/abs/2410.01553>, *arXiv:2410.01553*