Optimization of Large Language Models for Task Specific Applications

Aditya Kumar[1], Ranjit Singh[2], Manjit Singh[3]

{adityakr082@gmail.com[1], ranjit.bhatthal@gmail.com[2], manjit.behniwal@gmail.com[3]}

Department of Computer Science and Engineering, Chandigarh University, Mohali, India[123]

Abstract. The presence of the Large Language Models (LLMs) has been instrumental in automating tasks involving the complexities of language. Such tasks include the enhancement of human-computer interaction and the knowledge support provided across various fields. Growth in LLMs due to the advances in technology has resulted in personalized, scalable, and more efficient solutions in, among others, education, healthcare and entertainment areas. Through, LLMs have transformed, seventeen Natural Language Processing (NLP) getting to various different ways to customize this model of the task app. As a focal point of this work, we have four basic strategies: prompting engineering, fine-tuning, Low-Rank Adaptation (LoRA), Quantized LoRA (QLoRA), and Retrieval-Augmented Generation (RAG). These strategies along with their comparisons, pros, and cons, are especially treated in the text. The literature analysis and qualitive assessment in this study provide a basis for further research in the field of fast and efficient LLM adaptation. The main finding of the study is that full fine-tuning provides the best task-specific performance, even though it has high computation costs. On the other hand, LoRA and QLoRA are the best alternatives that are environmentally friendly since they reduce the required resources yet perform well. Prompt engineering and RAG clearly outperform in rapid deployment and dynamic information integration respectively.

Keywords: large language model, prompt engineering, fine-tuning, retrieval-augmented generation.

1 Introduction

The extraordinary expansion of large language models (LLMs), for example, GPT-3, BERT, and their up-to-date versions, has revolutionized numerous applications in natural language processing (NLP)[1]. On the other hand, the successful adjustment to tasks is still a major problem. Prompt engineering[2], full fine-tuning, and newer strategies such as LoRA[3], QLoRA[4], and retrieval-augmented generation (RAG)[5] are some of the techniques. Every

IICTDS 2025, December 19-20, Chandigarh, India
Copyright © 2026 EAI
DOI 10.4108/eai.19-12-2025.2363246

single one of them is dedicated to a particular condition of computational efficiency, model performance, and resource availability.

The remarkable success of large language models in tackling a variety of problems including, but not limited to, text generation, question answering, translation, and summarization, as they have done until now, marks the turning point in NLP[6][7][8]. The models that are pre-trained on huge collections of different data have revealed to be able to generalize over several fields and in this way, they have substantially decreased the need for task-specific models[9]. Nevertheless, LLMs when they are used on specialized tasks or areas require of course some adjustment to attain the best performance[10]. Numerous adaptation strategies have been developed to meet this challenge, and the adaptations each have their own benefits and drawbacks.

Despite the lack of a single best adaptation strategy, this manuscript will discuss the common base of five such solutions: prompt engineering, the whole possibility-fine-tuning, performance-efficient fine-tuning (LoRA and QLoRA), and retrieval-augmented generation (RAG). These methodologies show major divergences in both functioning principles and needs of resources and also in their adherence to specific tasks. The main purpose of this assessment is to bring out the advantages and disadvantages of each technique, giving users clear information on the conditions in which one approach may be more desirable than the other in real-world applications. This comparative assessment will furnish researchers and practitioners with a guideline for the most suitable adaptation method, because of their specific use sites.

We are going to clarify the scope of these methodologies, which is to say, their best-case applications. A systematic review will provide the researchers and practitioners with an unambiguous conception of the appropriate situations for the implementation of each method.

2 Background Study

The parallel growth in complexity and scale of LLMs by the demand for efficient and effective adaptation strategies is inevitable. Initial attempts to utilize pretrained models were mostly based on fine-tuning, a method which includes adjusting weights of the entire model to match a particular task or domain[11]. Although this method is useful, it can be quite costly in terms of computational power, particularly for the massive models, which require labor-intensive hardware resources and time[12].

As a result of these challenges, the development of alternative approaches has taken place, such as prompt engineering[13] and retrieval-augmented generation[5]. These concepts target to avoid or decrease retraining by innovatively requesting task-specific outputs from pretrained models. The low-rank adaptation methods[3] like LoRA and QLoRA[11] have recently come into existence, yielding performance-efficient tuning configurations that significantly lessen the traditionally associated with full model fine-tuning computational overhead[11].

All the above-mentioned techniques have their own limits and associated with different boundaries of flexibility, efficiency, and performance. A comprehensive knowledge of their limitations is a key factor to making optimal use of LLMs in real-life environments.

3 Existing Literature

This part accomplishes a comparative overview of the four methods of adaptation, departing from the existing literature. The evaluation is carried out based on the following main criteria: computational efficiency, adaptability to various tasks, memory consumption, and performance outcomes as reported in prior studies.

3.1 Prompt Engineering

Prompt engineering is the technique of creating particular input prompts in order to steer the output to the desired one from the LLM. In this way, since it does not change the model parameters, it is lightweight and can be used in applications where the computational resources are limited[2]. However, prompt engineering is sensitive to the variations of inputs, thus being not suitable for the cases in which a stable result is expected over its use.

It is a representation of a paradigm where model performance is driven by the careful construction of input prompts, rather than the modification of the model's weights. The inputs are carefully designed in a way that the model can reach the desired output without the need to alter its base parameters. An illustration of this is putting the model through specific instructions or examples that help the LLM simulate the template and therefore, output better-suited or more accurate answers[14]. Few-shot and zero-shot prompting are other strategies that have positively affected tasks when labeled data is limited or not available.

The starting point for the discussion is the belief that a well-structured input can yield the desired output from an LLM[14]. It has been proven that few-shot prompting, a process where only a limited number of examples are included in the prompt, can greatly increase model performance on specific tasks without having to undergo retraining. On the contrary, zero-shot prompting is reliant on the capability of the model to generalize without any task-specific examples being provided. This algorithm is appealing as it requires very few resources but is very risky because it is very sensitive to the way the prompt is built and thus it does not often generalize.

Although prompt engineering is a highly flexible and portable adaptation solution, it still struggles with some challenges such as the disposable nature of prompts and their being grounded on some model internal representations, like prompt sensitivity and pre-trained model inherent limitations, which exist since pre-training is done[15].

3.2 Full Fine-Tuning

In most cases, it is difficult to influence a model's inference about the unknown only by learning its prompt since it requires a lot of context. Though full fine-tuning constantly is the "go-to" approach model adaptation. A specific LLM's training can be continued for this purpose just like with anything else in the universe. By training a model on an appropriate dataset, this method brings the model closer to the final task. Generally, this technique is implemented by modifying all model parameters according to tasks, and only then the model is ready for downstream tasks. Thus, the performance also usually reaches the highest level in most tests. On various fields, it has been frequently used including natural language

processing and computer vision[16].

Full fine-tuning scales the whole parameter space, and as a result, the model gains the possibility to shift its internal representations into a more precise correlating form with the new data. Formally, given a pre-trained model with parameters θ_0 and the loss function $L(\theta)$ defined on the given task, full fine-tuning involves solving:

$$\theta' = \arg\min_{\theta} L(\theta) \tag{1}$$

Here, $\theta \approx \theta_0 - \eta \nabla_\theta L(\theta)$, and, η denotes the learning rate[17]. This update can give better performance when the target task is sufficiently complex or when the domain gap between the pre-training and target datasets is minimal.

Full fine-tuning is still the preferred choice for achieving task-specific results that particularly call for deep specialization. Nevertheless, its use is often impeded by the vastness of current LLMs. Fine-tuning the biggest of the models entails a very high amount of memory and computational power which are typically found in only the high-performance computing environments. Besides, the overfitting risk is particularly emphasized in small datasets that have a high requirement for proper tuning and regularization during the fine-tuning process[18]. This approach has high performance on the given task, although it is somewhat the opposite of high-efficiency, it indeed proves difficult because of resource requirements for fine-tuning, particularly for large models[19]. Overfitting is also a concern especially with small datasets. Full fine-tuning adjustments are always outside the reach of resources despite being effective, especially with large-scale LLM deployments.

3.3 LoRA and QLoRA

Owing to feasibility challenges faced by fine-tuning methods, one may consider the following two approaches that come at a lower cost.

LoRA and QLoRA are the newest methods for banking the computational expense of a full fine-tuning down. LoRA is a Low-Rank Adaptation that reduces the wastage of resources from full fine-tuning by only modifying parts of parameters in a lower-dimensional space [3].The method involves the low-rank updates of the model weights that are pre-trained facilitating the efficient task-specific adaptation without the need to change the entire model. LoRA replaces the full fine-tuning with a computationally effective method that is based on low-rank matrix updates. Instead of modifying the whole model's parameters, LoRA only changes a part of them which enables a task that is specific and tuned properly with comparatively less resources[20].

LoRA training, the type of training which it is, initiates from a base model while it preserves its weights in a frozen state. That means that the parameters of the pre-trained model stay unchanged[21]. QLoRA, a kind of an application of LoRA, goes a step further by using quantization methods, which lowers memory needs even more while still meeting the task-specific performance[4]. QLoRA, the one that is actually, is based on quantization and it reduces the memory that is used with no impact on the performance. This feature makes QLoRA an optimal choice for edge computing and resource-limited conditions[22], even though both techniques may be subject to performance losses on more comprehensive tasks.

QLoRA is initiated from a model that has undergone pre-training. Loading the model in the quantized format implies the model is not used in the full precision mode[23]. The standard choice is 4-bit quantization that is selected owing to its excellent balance vis-a-vis speed and accuracy performance. To the model's weights, QLoRA employs the 4-bit NormalFloat (NF4) quantization[4]. This step significantly reduces the memory footpring but also achieves the desired computational stability by keeping the activations (FP16, BF16, etc.) in a higher precision. Mathematically, for a quantized wieght matrix W, we have:

$$W_q = \text{Quant}(W) \tag{2}$$

where, W_q is the quantized representation, and dequantization is performed dynamically during inference. The process of adding LoRA adapters remains the same, except quantized weights are used.

After quantization, the weights are frozen and LoRA adapters are introduced in attention layer. Each LoRA adapter consists of two low-rank matrices A and B. Here, $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times d}$, with rank $r \ll d$, reducing the number of trainable parameters. The modified weight during fine-tuning is computed as:

$$W' = W_q + \Delta W \tag{3}$$

where, W_q is the quantized frozen weight and and ΔW is the LoRA adaptation. These methods have been especially useful in scenarios where computational resources are limited or where rapid adaptation is required. They are particularly useful for tasks with limited resources while maintaining strong performance.

3.4 Retrieval-Augmented Generation (RAG)

In cases when there is a large amount of frequently changing and/or unlabeled data, it becomes difficult to keep up with just the aforementioned methods.

Retrieval-Augmented Generation (RAG) constitutes a combination of an extrinsic retrieval feature and a generative model, thereby permitting the LLM to consult act-specific knowledge which is outside its internal database. These features are the ones that can be utilized to retrieve information related to them during the discovery phase of the project. In situations where the model's knowledge cutoff results in the use of outdated or irrelevant information, or when some information is really not there in the model's training phase, it is especially advantageous. Thus, it becomes possible for the model to introduce related external documents or datasets in real time. The approach is specifically useful in cases where the task is dependent on the real-time provision of the updated or specialized data[24].

RAG allows a large language model to access and generate information that is not in its weight by using an external database. This is the reason why RAG is the best choice for areas where timely and specific task knowledge is of utmost importance. Apart from that, there is also an issue with relying on retrieval systems, which can increase latency and also affect retrieval accuracy.

3.5 Hybrid Approaches

Despite the variety of approaches mentioned above that are single-solution pathways to a myriad of tasks, attention has also recently been focused on the coupling of several adaptation strategies so as to use all their potential synergies while at the same time diminishing their respective shortcomings. Combining strategies have remarkably been a very instrumental way in gaining a closure in the performance differential against application-specific fine-tuning and those strategies being more adjustable and fewer power-consuming decisions. Apart from the conventional use of hybrid approaches, their programming techniques, and their prevalence in the most recent literature, this piece also includes external programming techniques and their acceptance by the current literature.

3.5.1 Combining Prompt Engineering with External Retrieval

A prominent method that joins together retrieval-augmented generation and prompt engineering. This is a method that is able to adapt more of the prompt-based techniques through the integration of the external knowledge base. The studies demonstrate that the application of retrieval mechanisms with prompt engineering leads to big increases in response accuracy and the reduction of hallucination in response generation tasks[5]. This method supports the main idea of light prompts while opening the way to the use of either new or specific knowledge concerning the domain, thus it is very suitable for the use in dynamic, knowledge-intensive applications[25].

3.5.2 Fine-Tuning with Retrieval-Augmented Generation (FT RAG)

One more new tactic is the integration of full fine-tuning with retrieval-augmented generation, which makes it possible for models to adjust their internal representation and search through the external data sources dynamically. This approach is especially applicable in situations where full-fine tuning by itself is not enough to solve the quick changes in knowledge. For instance, investigations in the field of biomedical NLP have revealed that fine-tuning models in domain specific corpora and at the same time integrating RAG for real-time literature retrieval increases the performance of medical question answering tasks[24].

4 Comparative Analysis and Discussion

The approach selection of LLM adaptation is mainly based on the intricacy of the task, the resources available, and the requirements of the application. Although full fine-tuning is the benchmark for performance, it is not possible for big applications, due to it being resource-heavy. LoRA and QLoRA provide a balanced crossover of performance and computational efficiency making them suitable for cases where resources are limited but specific adaptation to a task is needed[3].

Although prompt engineering is of interest due to its simplicity, it is restricted by its inability to perform well on various tasks. RAG is a strong method for handling queries that pertain to a domain not typically encountered, but it might have problems, such as the time it takes to retrieve data and the need for external knowledge.

To distinctly define the methods, it is imperative to juxtapose them on fundamental theoretical parameters. The discussion will particularly address the aspects of computational efficiency, flexibility, resource requirements, and their potential for generalization to other tasks. The conclusions drawn from the analysis of existing literature will give an excellent basis for the understanding of appropriate timing and reasons to apply the methods.

Table 1. Comparison of methods

Aspect	Prompt Engineering	Fine-tuning	LoRA	QLoRA	RAG
Performance	Limited by base model's existing capabilities	High performance with significant task-specific improvements	Good balance of performance and efficiency	Similar to LoRA with minimal degradation	Excellent for knowledge-intensive tasks
Training Cost	No direct cost except prompt development time	Very high due to computational and data requirements	Moderate with reduced resource requirements	Low due to quantization benefits	Minimal - only knowledge base preparation
Running Cost	Standard API costs with longer prompts	Base inference plus model hosting costs	Slightly higher than base model	Similar to LoRA with efficient memory use	High due to vector database and retrieval costs
Computation	Base model inference only	Intensive GPU/TPU requirements	Moderate GPU requirements	Lower memory requirements than LoRA	Additional retrieval computation overhead
Accuracy	Highly variable based on prompt quality	High but risks catastrophic forgetting	Comparable to fine-tuning with better stability	Slight precision loss vs LoRA	High for factual queries within knowledge base

4.1 Computational Efficiency

When it comes to the adaptation of LLMs, efficiency is indeed a paramount factor. It is common knowledge that full fine-tuning is extremely computationally demanding particularly as the scale of the model increases[26]. For hefty models such as GPT-3, BERT, or T5 to make the adjustment of billions of parameters. obvious, they need to attach significant GPU or TPU resources. On top of that, fine-tuning is also a process that should be executed with caution, especially in relation to hyperparameters for avoiding the problem of overfitting, which is pertinent mainly when the target dataset is small. Conversely, LoRA and QLoRA are Federal Resource-efficient completely by using low-rank adaptations and quantization approaches respectively. Investigations have pronouncedly indicated that such techniques reduce both memory and compute costs without compromising performance significantly.

Prompt engineering, however, is the most efficient method among all since it requires no retraining at all. The only cost entailed in prompt engineering arises from the inevitable trial-and-error stage of coming up with prompts that work properly. Although this may result in a more effective use of computing machinery, the level of success achieved via prompt engineering is predominantly dependent on the task at hand, and the problem of creating

prompts that work across multiple tasks is still a hard one. Retrieval-Augmented Generation (RAG) can be considered a middle ground since it is the addition of an external retrieval system that will burden the periphery of the computational cost, particularly latency. On the other hand, a fair amount of the resources that are consumed by fine-tuning or training from scratch are countered by RAG's ability to deliver task specific information instantly.

Table 1 compares the stated methods with respect to cost, performance and usage.

4.2 Flexibility and Generalization

Flexibility is described as the model's ability to be adjusted to new tasks without affecting its native parts severely. Full fine-tuning gives the most flexibility since it allows to change the internal parameter of the model to fit the target task. Nevertheless, this very flexibility bears the price of generalization: An instance of a model modified for a specific task may have low performance on others without the additional retraining of the model.

Contrarily, prompt engineering is a method that may adapt tasks at a technical speed without the necessity to change the model's parameters. This characteristic renders it very much application-wide flexible, particularly in zero-shot and few-shot settings[27]. On the other hand, the efficacy of this technique is frequently determined by the prompt's quality, which acts as a source of variability and consequently affects generalization across tasks. LoRA and QLoRA are the intermediaries between the two mechanisms, prompt engineering, and overall fine-tuning being the extremes. After the implementation of low-rank adaptations, such methods would therefore task-specific fine-tuning and hold the majority of the pre-trained model generalization capabilities. The balance of detail and generality is the key reason that LoRA and QLoRA are favored when project criteria frequently alter.

Retrieval-Augmented Generation (RAG), which is retrieval-based model architecture, achieves generalization better through it using prediction of relevant external information[24]. In contrast to the previously suspended model's parameters, RAG is able to augment the model's knowledge with real-time data retrieval, which is why it is easily adaptable to new domains and arising information. Whereas, it is the quality of the retrieved information mostly affecting the model's performance, and still the optimal integration of retrieval into a generation framework is a challenge.

4.3 Resource Requirements

The resource requirements for different adaptation techniques are quite different. Complete fine-tuning is the adaptation method that needs the largest resources like hardware and time, especially working with big models. This stipulation makes it impractical for the companies which do not have sufficient computing power to use this approach.

Speaking of LoRA and QLoRA, these technologies not only provide a large array of resources but also cut costs by only tuning a part of the model's parameters. You can reduce memory usage effectively with LoRA which is a big advantage over full fine-tuning, whereas QLoRA, through quantization[28], achieves even greater reduction and thus is of particular importance for the edge devices and typical low-resource environments.

Prompt engineering, as per the theory, does not need any extra training resources other than

what is needed to do inference on a pre-trained model. Thus, it becomes the option requiring the least resources. However, the creation of good prompts is based on a lot of human efforts such as indirect costs. Although the generative model stays the same, the costs of using the retrieval system and processing the retrieved documents can differ greatly based on how difficult the task is and how extensive the external knowledge base is.

4.4 Scalability

The practical adaptation method's usage is determined by the scalability factor. Full fine-tuning is very full-featured but scales poorly with larger models, which implies that as the model size increases exponentially, the required resources also increase[29]. This definitely makes full fine-tuning an inefficient approach for organizations with limited computational resources or for large-scale applications.

On the other hand, LoRA and QLoRA are very scalable due to the fact that they introduce only slight alterations to the model's design. This is the reason why they can be effectively scaled even with LLMs. Furthermore, QLoRA's use of quantization also cuts down on the resource burden, thus it is especially attractive for cases that need large-scale deployment.

From a theoretical viewpoint, prompt engineering has infinite scalability; it does not require any retraining and it scales with no computational cost as the model increases in size. The difficulties that arise from the dependence on human intuition and trial-and-error at the stage of prompt construction make it a practical limitation for the scalability, especially for the more complicated tasks.

The retriveal system is the primary basis for the scalability of a RAG. Even though the generative part is mainly affected by the model, the retrieval part may be the one that if not optimized properly is dealing with the bottleneck. In real-world situations, the retrieval system needs to be optimally tuned in such a way as to deal with a large number of queries without any formidable latency or performance degradation.

4.5 Use Cases and Best Practices

Choosing the right adaptive method is greatly influenced by the certain needs of the task in question, the computational resources at hand, and the trade-in options of flexibility, performance, and scalability desired. The current section is a combinatorial source that brings together literature, and practical considerations to give the commonly accepted practices for selecting and implementing each approach.

4.5.1 Ideal Scenarios for Each Method

- *Prompt Engineering* is a method that finds its application primarily in instances where there is a restriction on computational resources and an immediate switch on to a different task is required. Under such conditions, it really works well in the absence of any kind of pre-trained data or with very few examples due to its ability to generalize learned knowledge to new problems. Typical tasks dealing with text classification or sentiment analysis where the categories are given beforehand are

good illustrations of this.

- *Full Fine-Tuning* is practical when the top-level performance of the task is the main concern, and resource issues are not of significant importance. Fine-tuning is suitable for tasks with plenty of labeled data and when the model requires a complete understanding of the target domain, such as medical diagnosis, legal text analysis, or specialized scientific research[30][31].
- The optimal application of *LoRA* and *QLoRA* can be seen in situations where resource efficiency is of utmost importance, despite the need for some level of task-specific fine-tuning. These techniques are especially effective in low-resource settings like mobile or edge devices, as well as during the adaptation of models for tasks that change very often.
- *RAG* performs best mainly when the task needs the instantaneous access of external knowledge or when the information of the pre-trained model is either not enough or outdated. RAG is mostly adopted in question-answering systems, dialogue agents, or any application where the domain knowledge is transient.

4.5.2 Recommendations for Choosing the Right Method

- For *general-purpose* NLP tasks that are flexible, minimize resource expenditure, prompt engineering is the top choice still.
- For the most *critical applications* which require the highest degree of accuracy and domain-specific, like healthcare or finance, full fine-tuning is recommended despite it is higher resource cost.
- *Resource-constrained* cases use LoRA and QLoRA as the top choices regarding performance and effectiveness, especially in the edge computing or mobile deployments.
- For *rapidly changing fields* where a knowledge base is constantly altered, RAG is the most powerful approach and can put real information from the environment into the model.

5 Conclusion

LLM adaptation is a field that is developing very quickly and this paper looks at a total of five dominant options which include- prompt engineering, full fine-tuning, LoRA, QLoRA, and retrieval-augmented generation- with the main point of the paper being to demonstrate the individual advantages of each alternative procedure based on factors such as task difficulty, the pool of resources, and the expected performance. Relying on prompt engineering is still the most lightweight and efficient, but the highest level of full fine-tuning corresponds to the greatest results in serious applications; LoRA and QLoRA are both options that avoid larger resource usage, while retrieval-augmented generation performs the best in new and information-centered environments. Although no single technique can be used in all cases, the analysis provided here can act as a roadmap when it comes to the choice of suitable methods, and the collaborative effort on hybrid models, automatic prompt design, and perpetual

learning would be the reasons for what should evolve further the skills and possibilities of LLM adaptation.

References

[1] M. Trajtenberg, “AI as the next GPT: A political-economy perspective,” National Bureau of Economic Research, 2018.

[2] J. White et al., “A prompt pattern catalog to enhance prompt engineering with chatgpt,” arXiv preprint arXiv:2302.11382, 2023.

[3] E. J. Hu et al., “Lora: Low-rank adaptation of large language models,” arXiv preprint arXiv:2106.09685, 2021.

[4] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, “Qlora: Efficient finetuning of quantized llms,” Advances in Neural Information Processing Systems, vol. 36, 2024.

[5] P. Lewis et al., “Retrieval-augmented generation for knowledge-intensive nlp tasks,” Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.

[6] Z. Liu et al., “Deid-gpt: Zero-shot medical text de-identification by gpt-4,” arXiv preprint arXiv:2303.11032, 2023.

[7] M. Leippold, “Thus spoke GPT-3: Interviewing a large-language model on climate finance,” Finance Research Letters, vol. 53, p. 103617, 2023.

[8] J. A. Alzubi, R. Jain, A. Singh, P. Parwekar, and M. Gupta, “COBERT: COVID-19 question answering system using BERT,” Arabian Journal for Science and Engineering, vol. 48, no. 8, pp. 11003–11013, 2023, doi: 10.1007/s13369-021-05810-5.

[9] G. Yenduri et al., “Gpt (generative pre-trained transformer)–a comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions,” IEEE Access, 2024.

[10] M. Gupta, R. Kumar, and T. Gautam, “Prediction of positive and negative sentiments for twitter data using machine learning,” in Proceedings of second international conference on computational electronics for wireless communications, S. Rawat, S. Kumar, P. Kumar, and J. Anguera, Eds., Singapore: Springer Nature Singapore, 2023, pp. 261–271.

[11] M. Wortsman et al., “Robust fine-tuning of zero-shot models,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 7959–7971.

[12] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, “Towards a unified view of parameter-efficient transfer learning,” arXiv preprint arXiv:2110.04366, 2021.

[13] L. S. Lo, “The CLEAR path: A framework for enhancing information literacy through prompt engineering,” The Journal of Academic Librarianship, vol. 49, no. 4, p. 102720, 2023.

[14] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, “A systematic survey of prompt engineering in large language models: Techniques and applications,” arXiv preprint arXiv:2402.07927, 2024.

[15] S. Tonmoy et al., “A comprehensive survey of hallucination mitigation techniques in large language models,” arXiv preprint arXiv:2401.01313, 2024.

[16] R. Jain, M. Gupta, S. Taneja, and D. J. Hemanth, “Deep learning based detection and

analysis of COVID-19 on chest x-ray images," Applied Intelligence, vol. 51, no. 3, pp. 1690–1700, 2021, doi: 10.1007/s10489-020-01902-1.

[17] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, Deep learning, vol. 1. MIT press Cambridge, 2016.

[18] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," arXiv preprint arXiv:2307.09288, 2023.

[19] H. W. Chung et al., "Scaling instruction-finetuned language models," Journal of Machine Learning Research, vol. 25, no. 70, pp. 1–53, 2024.

[20] S. Hayou, N. Ghosh, and B. Yu, "LoRA+: Efficient low rank adaptation of large models," CoRR, vol. abs/2402.12354, 2024, doi: 10.48550/ARXIV.2402.12354.

[21] Y. Zeng and K. Lee, "The expressive power of low-rank adaptation," in The twelfth international conference on learning representations, 2024. Available: https://openreview.net/forum?id=likXVjmh3E

[22] X. Zhang, N. Rajabi, K. Duh, and P. Koehn, "Machine translation with large language models: Prompting, few-shot learning, and fine-tuning with QLoRA," in Proceedings of the eighth conference on machine translation, 2023, pp. 468–481.

[23] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 2704–2713.

[24] S. Siriwardhana, R. Weerasekera, E. Wen, T. Kaluarachchi, R. Rana, and S. Nanayakkara, "Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering," Transactions of the Association for Computational Linguistics, vol. 11, pp. 1–17, 2023.

[25] X. Chen et al., "Decoupling knowledge from memorization: Retrieval-augmented prompt learning," Advances in Neural Information Processing Systems, vol. 35, pp. 23908–23922, 2022.

[26] T. Ding et al., "The efficiency spectrum of large language models: An algorithmic survey," arXiv preprint arXiv:2312.00678, 2023.

[27] J. Wang et al., "Prompt engineering for healthcare: Methodologies and applications," arXiv preprint arXiv:2304.14670, 2023.

[28] Z. Yao, R. Yazdani Aminabadi, M. Zhang, X. Wu, C. Li, and Y. He, "Zeroquant: Efficient and affordable post-training quantization for large-scale transformers," Advances in Neural Information Processing Systems, vol. 35, pp. 27168–27183, 2022.

[29] J. Kaplan et al., "Scaling laws for neural language models," arXiv preprint arXiv:2001.08361, 2020.

[30] S. Kaur and M. Singh, "Enhancing lung cancer diagnosis: A comprehensive analysis of machine learning and deep learning approaches," in Computational methods in science and technology, 1st ed., CRC Press, 2024, p. 4.

[31] A. Gupta, R. Kumar, and Y. Kumar, "An automatic speech recognition system in indian and foreign languages: A state-of-the-art review analysis," Intelligent Decision Technologies, vol. 17, no. 2, pp. 505–526, 2023, doi: 10.3233/IDT-220228.