CoRA: Optimizing Low-Rank Adaptation with Common Subspace of Large Language Models

Anonymous ACL submission

Abstract

Parameter-efficient fine-tuning (PEFT) methods are crucial for adapting pre-trained models to downstream tasks while avoiding additional inference costs. Among these, Low-Rank Adaptation (LoRA) is widely used. However, it still exhibits an accuracy gap compared to full fine-tuning (FT), possibly due to redundant parameters in its trained matrices. In this work, we present a novel approach to reducing the proportion of redundant parameters, with the aim of narrowing the accuracy gap between LoRA and FT. To investigate for the reduction of redundant parameters, we propose Commone-subspace Low-Rank Adaption(CoRA) as a promising approach. CoRA extracts the common basis matrices from the LLMs in order to obtain higher quality parameters, subsequently using them by LoRA to adapt the Downstream tasks, so that more effective parameters can be applied to the model output, despite a limited budget for the parameters. The experimental results demonstrate that CoRA achieves competitive performance compared to PEFT methods such as LoRA and reduces the accuracy gap with FT.

1 Introduction

Pre-trained models built on extensive datasets have shown strong generalization abilities, driving significant advancements across a variety of fields, including natural language processing (NLP) (Qiu et al., 2020; Min et al., 2023) and multimodal tasks (Gupta et al., 2023; Wang et al., 2023a). However, as large language models (LLMs) continue to grow in size (Ding et al., 2023b), comprehensive fine-tuning (FT) becomes increasingly impractical for many downstream applications (Unite.AI, 2023). The large number of parameters in such models often includes redundant components (Men et al., 2024; Chiang and Lee, 2024), and minimizing the optimization of these non-essential parameters is desirable to improve training efficiency and practicality (Lee et al., 2024).

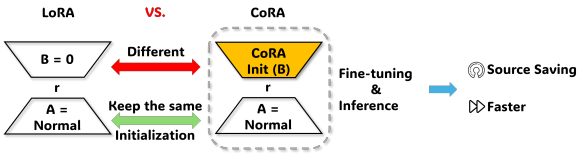

Figure 1: A simple example can illustrate the issues addressed by this paper and the role of CoRA, which involves replacing the initialization of the B matrix in LoRA with a common basis matrix tocomplete the fine-tuning tasks.

According to (Wu et al., 2023), the success of models does not rely solely on the full scope of information captured by the pretrained model. Instead, it hinges on the alignment between the structural requirements of the downstream task and the characteristics of the representations learned during pretraining. Similarly, (Wei et al., 2021a) demonstrate that the knowledge embedded in pretrained language models facilitates downstream tasks by capturing the latent generative structure of text. In this framework, the downstream classifier is tasked with recovering pertinent latent information from this underlying structure to solve the specific task at hand. It can be inferred that downstream models trained on the same pretrained model, while incorporating task-specific knowledge extracted from the original model, still share a substantial amount of common knowledge. By leveraging this shared knowledge, it is possible to mitigate redundant parameters, leading to more efficient models.

Analyzing the trained downstream models, we observed that shared knowledge within them can optimize parameter quality, especially where redundant parameters exist. To leverage this, we propose constructing parameter matrices by extracting corresponding matrices across different downstream models, facilitating the identification of shared knowledge. Our experiments reveal a

potential common subspace among these models, termed the common basis matrix, which contains more efficient parameters. To address the large number of parameters in the common basis matrix, we compress it and then enhance its utility through PEFT (Houlsby et al., 2019). In this process, Low-Rank Adaptation (LoRA) (Hu et al., 2021) has demonstrated notable performance due to its simplicity and retention of the model architecture. Building on this, we introduce Common-subspace Low-Rank Adaptation (CoRA), which replaces LoRA's matrices with the extracted common basis matrix for fine-tuning, as depicted in Figure 1. This approach maintains LoRA's strengths while further enhancing parameter fine-tuning efficiency.Compared to LoRA, CoRA shows significant improvements on both the AlpacaClean and CodeToText datasets, with the most notable gains in SacreBLEU and METEOR scores. On the AlpacaClean dataset, CoRA achieves an 18.55% increase in SacreBLEU and a 9.22% increase in METEOR. On the CodeToText dataset, SacreBLEU improves by 12.10% and METEOR by 10.00%. Additionally, CoRA sees a 5-10% improvement in the ROUGE metrics. In commonsense reasoning tasks, CoRA also consistently outperforms LoRA and LoRA+, and when training LLaMA2-7B on the HellaSwag dataset, it surpasses both training methods of DoRA. Compared to LoRA, CoRA demonstrates significant improvements across both the AlpacaClean and CodeToText datasets, with the most notable advancements observed in SacreBLEU and METEOR scores. Furthermore, CoRA achieves a 5-10% enhancement in ROUGE metrics. In commonsense reasoning tasks, CoRA consistently outperforms both LoRA and LoRA+, and when applied to training LLaMA2-7B on the HellaSwag dataset, it surpasses both DoRA training methods The summary of our contributions is as follows:

- We introduce a novel approach, CoRA, which leverages the existence of a common subspace among large models to enhance their learning capabilities. Experimental results indicate that CoRA's training performance closely matches that of FT and exceeds the capability of LoRA.

- CoRA identifies and successfully extracts the common basis matrix among large models. Experimental evidence suggests that this common basis matrix could reduce the proportion of redundant parameters, thereby enhancing model stability and improving output quality.

- On the experimental datasets, which included NLP tasks involving common-sense reasoning and other areas, CoRA demonstrated superior performance compared to LoRA.

2 Related Works

2.1 Parameter-Efficient Fine-Tuning

The exponential growth in parameters in Transformer-based pre-trained language models (PLMs) (Vaswani et al., 2017), especially in large models like LLaMA (Touvron et al., 2023), has significantly advanced numerous NLP tasks. This increase in complexity has enhanced model effectiveness, particularly when fine-tuned for specific downstream datasets. Unlike zero-shot learning (Wei et al., 2021b), fine-tuning allows models to better capture task-specific nuances, leading to superior performance.

However, scaling these models presents major challenges due to the substantial computational resources required for fine-tuning. To address this, PEFT (Ding et al., 2023a; Liu et al., 2023) have been developed. Approaches such as Prompt Tuning (Shin et al., 2020; Chen et al., 2022) and Adapter Tuning (Houlsby et al., 2019; Newman et al., 2021) minimize the need for extensive fine-tuning by using task-specific prompts or integrating a small set of specialized parameters, thereby reducing resource consumption while maintaining comparable performance. Despite these advances, models may still underperform in new domains with unseen entity types. Prefix Tuning (Li and Liang, 2021; Vos et al., 2022) offers an alternative by optimizing a small set of task-specific vectors, facilitating efficient token processing with minimal computational overhead. These methods offer practical strategies for deploying large NLP models in resource-constrained environments (Tabani et al., 2021).

Although all the aforementioned methods have markedly decreased the computational requirements, fine-tuning these extensive models still demands significant resources, especially in environments with limited computational capacity (Liao et al., 2023). Therefore, even parameter-efficient fine-tuning techniques can lead to prolonged computation times in low-resource settings such as those involving consumer-grade graphics cards.

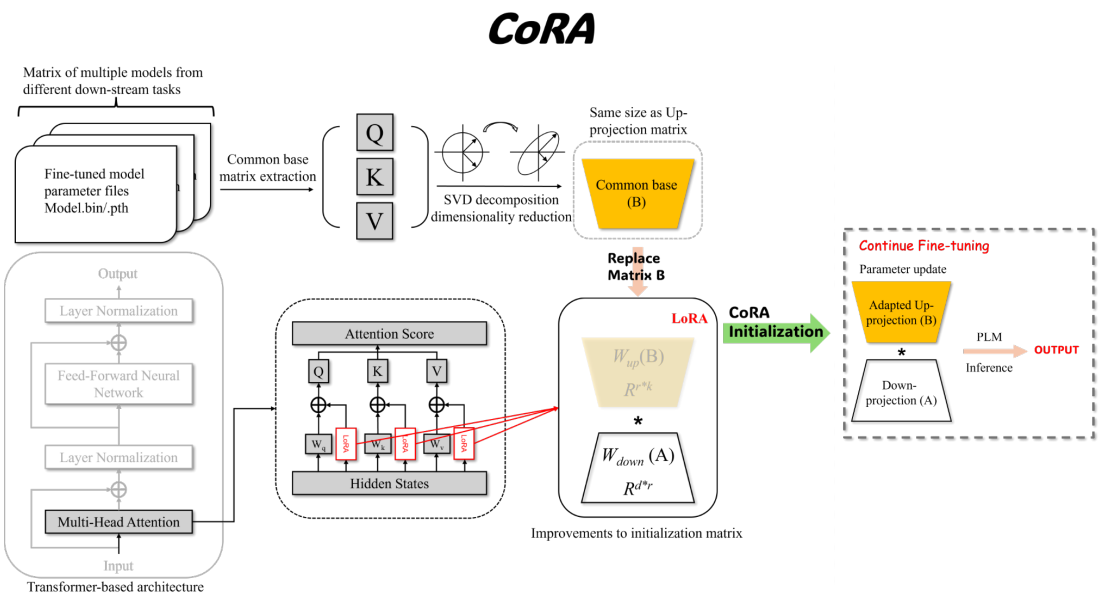

Figure 2: An overview of CoRA. In the downstream large models, we extracted a common basis matrix within the corresponding attention heads' Q, K, and V matrices. Utilizing Singular Value Decomposition (SVD) for dimensionality reduction, we adapted this matrix to meet the projection specifications required by the LoRA fine-tuning paradigm. The adapted common basis matrix replaced the original B matrix in the LoRA on the corresponding attention heads' Q, K, and V matrices. This modification was integrated into subsequent training processes then continue do fine-tuning.

2.2 Low-Rank Adaption of LLMs

Low-Rank Adaptation (LoRA) (Hu et al., 2021) represents a transformative approach in fine-tuning pre-trained deep learning models. This method enhances the model's attention mechanism by incorporating low-rank up-projection and down-projection matrices. It facilitates model adaptation to new tasks or datasets by targeting a minimal subset of parameters for adjustment, thereby maintaining the integrity of the original model's parameters. Its design prioritizes efficiency and flexibility, reducing training and inference times without compromising performance. However, a notable challenge with LoRA arises in scenarios with limited computational resources, such as edge computing devices, where it is difficult to balance model accuracy with computational and memory constraints. Numerous variants of LoRA have been developed, each offering distinct enhancements to the fine-tuning process. One notable variant, QLoRA (Dettmers et al., 2023; Wang et al., 2023b), employs 4-bit quantized backpropagation combined with Low-Rank Adapters to facilitate the efficient training of large models, accommodating up to 65 billion parameters on a single 48GB GPU. This variant introduces several technological advancements, such as the 4-bit NormalFloat data type, Double Quantization, and Paged Optimizers, designed to optimize memory utilization without degrading model performance. Another innovative variant, QA-LoRA (Xu et al., 2023), builds upon the LoRA framework by incorporating quantization-aware techniques that enhance memory and computational efficiency during fine-tuning. It introduces group-wise operators for refined quantization and adaptation, effectively merging the advantages of LoRA with significant improvements in computational efficiency, which is particularly beneficial in resource-limited environments. Furthermore, DyLoRA (Valipour et al., 2022) tackles the rigidity of traditional LoRA approaches by enabling training across a spectrum of ranks. This flexibility accelerates the training process 4 to 7 times, depending on the specific task, and maintains high performance across diverse pre-trained models like RoBERTa (Liu et al., 2019), and GPT (Floridi and Chiriatti, 2020).

Recent advancements have seen a proliferation of sophisticated approaches to optimize LoRA, such as DoRA (Liu et al., 2024), which proposes a low-rank, weight decomposition for adaptability. This method splits the pre-trained weights into two components—magnitude and direction—to facilitate fine-tuning. Specifically, it uses LoRA to

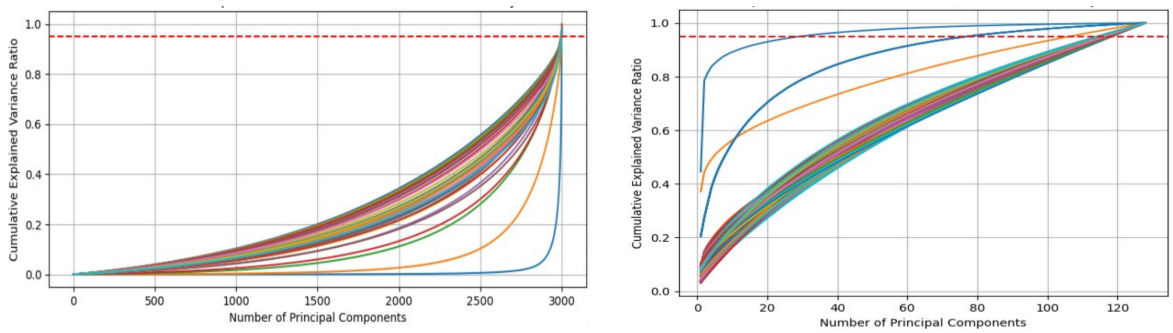

Figure 3: The number of principal components or singular values required to explain the variance using PCA (left) and SVD (right) techniques fully. PCA requires about 3000 principal components, while SVD only requires about 130 singular values, showing the efficiency of SVD in dimensionality reduction of high-dimensional data.

update the direction, effectively minimizing the number of trainable parameters. By employing DoRA, the learning capability and training stability of LoRA are enhanced without adding extra inference overhead. Furthermore, LoRA+ (Hayou et al., 2024) introduces different learning rates for the LoRA adapter matrices A and B, correcting suboptimal aspects of LoRA through carefully chosen ratios. This adjustment boosts performance and fine-tuning speed, achieving a twofold acceleration with the same computational cost as LoRA. Considering the potential limitations of low-rank updates in restricting the effective learning and memory capabilities of large language models (LLMs), MoRA (Jiang et al., 2024) has introduced the use of square matrices for high-rank updates, maintaining the same number of trainable parameters. It incorporates non-parametric operators to reduce the input dimensions of the square matrix while increasing the output dimensions. These operators ensure that the weights can be integrated back into the LLM, resulting in superior performance on memory-intensive tasks and comparable outcomes on others with LoRA.

3 Method

3.1 Workflow of The Entire Framework

The entire experiment is based on Llama-7B, Llama-13B, Llama2-7B and Llama2-13B, including the complete attention head mechanism suitable for our related experiments. We can obtain each layer's Q, K, and V matrices (Vaswani et al., 2017) from different downstream task models and put all of them into one matrix $W_0 = [W_Q, W_K, W_V]^T \in \mathbb{R}^{3d_{model} \times d_k}$. Referring to the technical principles of LoRA, to effectively adjust the pre-trained Transformer model, LoRA targets the critical weight matrices Q, K, and V. Due to the unique construction of various models, additional matrices are required during training. For instance, the Feed-Forward Network Layer (FFN) in the Llama architecture includes up and $down$ matrices, which serve to expand and compress data dimensions, respectively. These operations are particularly critical in the FFN segment of the transformer model, necessitating their simultaneous training. These matrices usually contain many parameters, and LoRA combines them into two matrices with lower rankings (denoted $A \in \mathbb{R}^{3d_{model} \times r}$ and $B \in \mathbb{R}^{r \times d_k}$, and $r \ll min(d_{model}, d_k)$) to reduce the number of parameters.

$$W = W_0 + AB, \tag{1}$$

In our approach, $W_0 = [W_Q, W_K, W_V]^T$, where r is the target dimension of the word vector in the LoRA dimensionality reduction process, as the Figure 2 shown. This low-rank approximation significantly reduces parameter scale and lowers computational and storage demands. Because the function of A is to update enhance adaptability, we chose common basis matrices as B's initialization . We employ r as the target dimension in our matrix dimensionality reduction process. By using SVD, the dimension of the common basis matrix is aligned with the size of r. This matrix is derived as:

$$W_0 = \frac{1}{n} \sum_{i=1}^{n} W_i = [W_{Q_i}, W_{K_i}, W_{V_i}]^T \tag{2}$$

The following section will discuss the experimental basis for selecting this strategy.

$$W_0 = U\Sigma V^T \tag{3}$$

$$so \qquad V^T = \Sigma^{-1} U^T W_0 \tag{4}$$

We obtained the three data variables ($U \in \mathbb{R}^{3d_{model} \times 3d_{model}}, \Sigma \in \mathbb{R}^{3d_{model} \times d_k}, V^T \in \mathbb{R}^{d_k \times d_k}$) on the right through W_0 decomposition, where V^T is the transpose of the right singular vector matrix V, and we take the first r as the dimensionality reduction representation of the common space matrix.

$$W_{reduced} = B = V^T[: r, :] \tag{5}$$

$$W = W_0 + AW_{Reduced} \tag{6}$$

After replacing matrix B, Involving matrix A in training to capture information relevant to new tasks and improve performance in downstream tasks and enhance the model's overall performance.

3.2 Methods For Matrix Compression Extraction

In feature extraction and data dimensionality reduction, Principal Component Analysis (PCA) (Maćkiewicz and Ratajczak, 1993) and Singular Value Decomposition (SVD) (Abdi, 2007) are pivotal technologies. A thorough analysis of these methodologies provides insight into their application strengths and delineates their performance boundaries.

PCA identifies directions of maximum variance in the data, termed principal components, by eigendecomposition of the covariance matrix Σ, where the covariance matrix is given by:

$$\Sigma = \frac{1}{3d_{model} - 1} \sum_{i=1}^{n} (W_0 - \mu)(W_0 - \mu)^T \tag{7}$$

$$\Sigma = PDP^{-1} \tag{8}$$

$$W'_{reduced} = W_0 \cdot P[:, 1 : r] \tag{9}$$

Here, W_0 represents the matrix formed by merging the Q, K, and V matrices, with μ as the mean of all data points. The principal components are the eigenvectors of the covariance matrix Σ, ordered by their eigenvalues. The matrix P contains these eigenvectors, and we select the first r for dimensionality reduction. This process simplifies the data while retaining most of the essential information.

On the other hand, SVD, a versatile decomposition method for both square and rectangular matrices, reveals the fundamental structure of data. In dimensionality reduction, SVD efficiently compresses data by retaining only the largest singular values and associated singular vectors. Formula (3)(4)(5) outline the SVD process in this task. Our experiments confirmed its effectiveness in our work.

For the Q, K, and V matrices across all layers of the Llama2 fine-tuned model, we applied both SVD and PCA to reduce the dimensionality of high-dimensional data and observed the changes in real-time interpretation rates. As shown in Figure 3, achieving 100% explained variance with PCA requires around 3000 components, whereas SVD achieves this with just 130 singular values.

Thus, informed by these comparative results, we selected SVD as the dimensionality reduction method for our CoRA lightweight fine-tuning strategy. SVD not only drastically reduces the number of components but also ensures full variance explanation, making it particularly advantageous for large datasets by effectively compressing data, reducing computational costs, and accelerating model training and prediction.

3.3 Selection of Replacement Matrices

To determine how many models are required to extract a common basis matrix to serve as the B matrix, we randomly selected 1 to 5 models from a pool of 5 for our experiments. This random selection ensured fairness and minimized bias. The models were sourced from different downstream tasks to investigate the possibility of extracting and utilizing a common basis matrix.

Based on our experimental results, as illustrated in Table 3.2, it is clear that even when the common basis matrices are updated, the performance remains stable across both the alpaca clean (Ji, 2024) and text-to-code datasets. This indicates that, following fine-tuning for downstream tasks, large models preserve a shared knowledge space—referred to as the common basis matrix in this paper—which ensures consistent results, irrespective of the number of matrices extracted.

The results indicate that any extracted matrix from the table could be used for future experiments, although some may be less efficient. To ensure accuracy, we selected the matrix from the five models that demonstrated the best and most consistent overall performance for further experimentation.

Table 1: Evaluation of the performance of LlaMa2-13B, fine-tuned with optimization on downstream tasks using common basis matrices extracted from varying numbers of models on the $AlpacaCleaned$ dataset and $Text-to-Code$ dataset. The table below lists the inference results assessed by the evaluation methods. The target matrix is just Q, K, V.

Method	Dataset	Numbers	ROUGE-1	ROUGE-2	ROUGE-L	METEOR Score	SacreBLEU Score	BERT Score
CoRA	AlpacaClean	1	0.419	0.231	0.394	0.322	17.184	0.898
CoRA	AlpacaClean	2	0.418	0.230	0.395	0.319	16.943	0.897
CoRA	AlpacaClean	3	0.424	0.231	0.399	0.321	17.093	0.898
CoRA	AlpacaClean	4	0.424	0.231	0.399	0.322	17.063	0.898
CoRA	AlpacaClean	5	0.421	0.230	0.396	0.320	17.296	0.897
CoRA	CodeToText	1	0.414	0.252	0.404	0.327	27.692	0.894
CoRA	CodeToText	2	0.413	0.248	0.403	0.326	27.461	0.894
CoRA	CodeToText	3	0.413	0.250	0.403	0.326	27.814	0.894
CoRA	CodeToText	4	0.409	0.246	0.400	0.324	27.180	0.893
CoRA	CodeToText	5	0.418	0.255	0.407	0.330	28.874	0.894

Table 2: Performance evaluation of downstream tasks on the $AlpacaCleaned$ dataset and $Text-to-Code$ dataset, following fine-tuning with optimization using common basis matrices, full parameter fine-tuning, and LoRA fine-tuning. The table below lists the inference results assessed by the evaluation methods. The target matrix is just Q, K, V.

Method	Dataset	ROUGE-1	ROUGE-2	ROUGE-L	METEOR Score	SacreBLEU Score	BERT Score
FineTuning	AlpacaClean	0.359	0.170	0.336	0.273	11.159	0.887
LoRA	AlpacaClean	0.397	0.214	0.374	0.293	14.587	0.893
CoRA	AlpacaClean	0.421	0.230	0.396	0.320	17.296	0.897
FineTuning	CodeToText	0.463	0.355	0.456	0.400	23.744	0.885
LoRA	CodeToText	0.393	0.231	0.381	0.300	25.758	0.889
CoRA	CodeToText	0.418	0.255	0.407	0.330	28.874	0.894

4 Experiments

We conducted our experiments using the LLaMA series as the backbone large language model. For training downstream tasks, we used datasets including Alpaca Clean(Taori et al., 2023), a text-to-multi-language (text-to-code) programming dataset from Hugging Face, and commonsense reasoning datasets(Hu et al., 2023a), along with a custom text-to-code dataset. To validate our experimental results, we employed a comprehensive set of evaluation metrics: ROUGE-1, ROUGE-2, ROUGE-L(Lin, 2004), METEOR(Denkowski and Lavie, 2014), SacreBLEU(Keenan, 2017), BERT Score(Zhang et al., 2019), and Accuracy. These metrics evaluated text similarity, translation quality, semantic similarity, and overall accuracy, providing a thorough assessment of model performance. The training process was conducted on NVIDIA® A100 GPUs with 80GB of memory.

4.1 Results of The Main Experiment

The comparison results of FT, LoRA fine-tuning and CoRA fine-tuning under the same rank are recorded in Table 3.2, which indicate that the CoRA outperformed the LoRA on two datasets. In the alpaca clean dataset, compared to the FT, CoRA surpasses them significantly in all evaluation method. And the performance of CoRA in evaluating the text-to-code dataset demonstrated considerable advantages as well . Although it does not fully outperform FT in terms of scoring metrics, it still demonstrates its capbilities over LoRA. These results confirm its adaptability and efficiency in complex generation tasks.

The results obtained from the experiments validate that leveraging a common basis matrix in CoRA fine-tuning greatly enhances the model's ability to learn and generalize. This approach allows for a more efficient utilization of parameters, leading to superior performance in downstream tasks compared to traditional LoRA fine-tuning. The observed improvements further suggest that because the common basis matrix maintains a closer alignment with the knowledge encapsulated in the pre-trained model, the model is better able to leverage this knowledge for downstream tasks. This

Table 3: Accuracy comparison of LLaMA 7B/13B and LLaMA2 7B with various PEFT methods on seven commonsense reasoning datasets. Results of all the baseline methods on LLaMA 7B/13B are taken from (Hu et al., 2023b) and DoRA results are taken from (Liu et al., 2024). Results of LoRA on LLaMA2 7B are obtained using the hyperparameters described in (Hu et al., 2023b). DoRA†: the adjusted version of DoRA with the rank halved.

Model	PEFT Method	# Params (%)	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA
ChatGPT	-	-	85.4	68.5	78.5	66.1	89.8	79.9	74.8
LLaMA-7B	Prefix	0.11	76.8	73.9	42.1	72.1	72.9	54.0	60.6
	Series	0.99	79.2	76.3	67.9	75.7	74.5	57.1	72.4
	Parallel	3.54	76.4	78.8	69.8	78.9	73.7	57.3	75.2
	LoRA	0.83	80.7	77.4	78.1	78.8	77.8	61.3	74.8
	LoRA+	0.83	79.3	76.5	79.3	77.8	75.2	58.1	68.4
	CoRA	0.83	81.1	78.6	85.8	79.0	81.6	63.4	77.4
	DoRA†	0.43	82.6	79.7	83.2	80.6	80.6	65.4	77.6
	DoRA	0.84	83.4	78.6	87.2	81.0	81.9	66.2	79.2
LLaMA-13B	Prefix	0.03	75.4	72.1	55.2	68.6	79.5	62.9	68.0
	Series	0.80	83	79.2	88.1	82.4	82.5	67.3	81.8
	Parallel	2.89	84.9	79.8	92.1	84.7	84.2	71.2	82.4
	LoRA	0.67	83.5	80.5	90.5	83.7	82.8	68.3	82.4
	LoRA+	0.67	83.7	80.3	90.3	83.8	82.9	68.5	82.3
	CoRA	0.67	85.1	80.9	90.7	83.5	83.2	68.6	82.8
	DoRA†	0.35	85.3	79.9	90.1	82.9	82.7	69.7	83.6
	DoRA	0.68	84.9	81.5	92.4	84.2	84.2	69.6	82.8
LLaMA2-7B	LoRA	0.83	79.9	79.5	83.6	82.6	79.8	64.7	81.0
	LoRA+	0.83	79.6	79.1	86.1	81.8	80.4	66.7	80.7
	CoRA	0.83	83.7	78.8	91.0	82.3	84.2	69.8	79.6
	DoRA†	0.43	83.1	79.9	89.1	83.0	84.5	71.0	81.2
	DoRA	0.84	83.7	76.0	89.1	82.6	83.7	68.2	82.4

consistency minimizes the risk of overfitting to specific tasks while also preventing the dilution of pre-trained knowledge, a common issue in traditional fine-tuning methods.

Additionally, the common basis matrix contributes to a more optimized allocation and use of parameters, reducing the redundancy typically seen in standard fine-tuning. This optimization ensures that more key parameters are directly involved in producing the model's outputs, enhancing the model's adaptability across different tasks. Consequently, this method achieves results that either match or surpass those of full fine-tuning, particularly in complex generation tasks. These findings highlight CoRA fine-tuning as a more resource-efficient alternative, not only preserving the core competencies of pre-trained models but also enabling strong adaptability to task-specific requirements, making it an effective approach for optimizing large language models in various downstream applications.

4.2 Comparison of CoRA with Other PEFT Methods

In comparison with Prefix, Series, and Parallel methods, CoRA has demonstrated significant advantages. Furthermore, CoRA outperforms LoRA in nearly all tested models, with the exception of a slightly lower score on the OBQA dataset in the LLaMA2-7B model, which may be attributed to errors during the model training process.

In comparisons with LoRA+, CoRA also exhibits superior performance. For instance, in the PIQA task of the LLaMA2-7B model, CoRA achieved a score of 83.7, compared to just 79.6 for LoRA+. Additionally, in the HellaSwag task, CoRA surpassed LoRA+ by 4.9. Overall, CoRA demonstrates a stronger capability in understanding and applying extensive contextual information than LoRA+.

In the ARC and HellaSwag tasks, CoRA demonstrates more pronounced advantages compared to DoRA. Both tasks involve complex reasoning and commonsense retrieval, requiring the model to effectively draw on the knowledge stored in the pre-trained model. CoRA excels in leveraging shared information from the pre-trained model, reducing redundant parameters, and enhancing the efficiency and accuracy of reasoning. The ability to integrate global knowledge is crucial in these tasks, and this is where CoRA holds a significant advantage over DoRA. In contrast, while DoRA's decomposition

strategy allows for more precise parameter adjustments, it may hinder the model's ability to capture the broader knowledge necessary for tasks requiring large-scale knowledge integration, thus affecting its reasoning performance. As a result, CoRA demonstrates superior performance in certain tasks, highlighting its competitiveness against state-of-the-art (SOTA) methods.

In summary, CoRA not only excels in individual tasks but also demonstrates significant superiority over methods like Prefix, Series, Parallel, and LoRA+ across various models. This highlights CoRA's stability and adaptability in multi-task and multi-model environments, reinforcing its position as a competitive approach against SOTA techniques.

5 Analysis

To objectively and impartially evaluate the performance of the Generation model, we employed GPT-4 (Achiam et al., 2023) to score the model's outputs in conjunction with the original dataset results. Our assessment focuses on three aspects: Fluency, Relevance, and Accuracy (Yang et al., 2024). We aim to determine whether the model's outputs are semantically fluent, whether the content aligns with human interpretative expectations, and how relevant the generated results are to the intended outcomes. Additionally, accuracy is assessed by examining the consistency of the model's outputs with the original data. This set of evaluation criteria is designed to ensure that the model's outputs meet our expected standards.

The specific results are in the Table 5. The objective evaluation of GPT-4 shows that all two models achieved the expected scores. This demonstrates that our approach fulfilled the experiment's objectives and ensured the utility of our Generation model's outputs.

Table 4: Scoring Results: Evaluating model outputs' fluency, relevance, and accuracy using GPT-4.

Dataset	Average Fluency	Average Relevance	Average Accuracy
CodeToText	88.90	81.91	82.79
AlpacaClean	90.39	83.90	83.92

Additionally, utilizing a text-to-code dataset, we manually reviewed and executed the generated code to assess its readability and correctness. The scoring outcomes are detailed in Table 5 as follows:

Upon evaluating and summarizing, we find that the code's clarity and ease of understanding are high. This indicates that the generated code has a logical structure, follows naming conventions, and is well-commented, allowing our reviewers to comprehend the logic and functionality of the code easily.

Table 5: Evaluation Results: Manual inspection of code with scoring based on readability and accuracy.

Dataset	Average Readability	Average Correctness
CodeToText	87.72	90.30

The high accuracy score signifies that the code achieves the intended functions and closely matches the specification requirements in execution. This generally means that most of the generated code performs stably across various test scenarios, has a low error rate, and can reliably complete the tasks it was designed to perform.

6 Conclusion

We propose utilizing the common subspace of a large model to effectively reduce redundant parameters, demonstrating how to achieve better performance in large language models while using the same computational resources. By leveraging a shared common basis matrix, we enhance the model's ability to learn downstream tasks, maintain fine-tuning effectiveness, and surpass the performance of traditional LoRA and other commonly used PEFT methods. In some cases, we even exceed the state-of-the-art results of DoRA. These findings highlight the importance of considering computational efficiency in model design and optimization, offering a practical strategy for training large-scale language models in resource-constrained environments. In the future, we plan to explore further method to optimize common basis matrix so that it could be to train larger models with fewer parameters, enabling them to better leverage their knowledge base for specific downstream tasks.

7 Limitations

Our experiments have certain limitations. First, the proposed method has only been applied to the Llama series of LLMs and tested exclusively on tasks in the NLP domain, including comparative experiments and generalization tests. While the results demonstrate effectiveness in NLP tasks, we have not yet validated the method in other domains, such as computer vision (CV) and graph neural

networks (GNN). This represents an area for further improvement in future work.

References

Hervé Abdi. 2007. Singular value decomposition (svd) and generalized singular value decomposition. *Encyclopedia of measurement and statistics*, 907(912):44.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web conference 2022*, pages 2778–2788.

Cheng-Han Chiang and Hung-yi Lee. 2024. Over-reasoning and redundant calculation of large language models. *arXiv preprint arXiv:2401.11467*.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2023a. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Qinxu Ding, Ding Ding, Yue Wang, Chong Guan, and Bosheng Ding. 2023b. Unraveling the landscape of large language models: a systematic review and future perspectives. *Journal of Electronic Business & Digital Economics*, (ahead-of-print).

Luciano Floridi and Massimo Chiriatti. 2020. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694.

Kshitij Gupta, Benjamin Thérien, Adam Ibrahim, Mats L. Richter, Quentin Anthony, Eugene Belilovsky, Irina Rish, and Timothée Lesort. 2023. Continual pre-training of large language models: How to (re)warm your model? *Preprint*, arXiv:2308.04014.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Zhiqiang Hu, Yihuai Lan, Lei Wang, Wanyu Xu, Ee-Peng Lim, Roy Ka-Wei Lee, Lidong Bing, and Soujanya Poria. 2023a. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023b. LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Peiqi Ji. 2024. Text sentiment analysis based on llama models. In *International Conference on Computer Graphics, Artificial Intelligence, and Data Processing (ICCAID 2023)*, volume 13105, pages 595–600. SPIE.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, et al. 2024. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*.

William JF Keenan. 2017. Sacre bleu: faith, fashion and freedom: Marist foundation garments 1817–1862. In *Materializing Religion*, pages 132–153. Routledge.

Younghun Lee, Sungchul Kim, Ryan A. Rossi, Tong Yu, and Xiang Chen. 2024. Learning to reduce: Towards improving performance of large language models on structured data. *Preprint*, arXiv:2407.02750.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Baohao Liao, Shaomu Tan, and Christof Monz. 2023. Make your pre-trained model reversible: From parameter to memory efficient fine-tuning. *arXiv preprint arXiv:2306.00477*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. Gpt understands, too. *AI Open*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrzej Maćkiewicz and Waldemar Ratajczak. 1993. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2024. Shortgpt: Layers in large language models are more redundant than you expect. *Preprint*, arXiv:2403.03853.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40.

Benjamin Newman, Prafulla Kumar Choubey, and Nazneen Rajani. 2021. P-adapters: Robustly extracting factual information from language models with diverse prompts. *arXiv preprint arXiv:2110.07280*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China technological sciences*, 63(10):1872–1897.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Hamid Tabani, Ajay Balasubramaniam, Shabbir Marzban, Elahe Arani, and Bahram Zonooz. 2021. Improving the efficiency of transformers for resource-constrained devices. In *2021 24th Euromicro Conference on Digital System Design (DSD)*, pages 449–456. IEEE.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Unite.AI. 2023. A full guide to fine-tuning large language models. Accessed: 2024-10-12.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2022. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

David Vos, Till Döhmen, and Sebastian Schelter. 2022. Towards parameter-efficient automation of data wrangling tasks with prefix-tuning. In *NeurIPS 2022 First Table Representation Workshop*.

Xiao Wang, Guangyao Chen, Guangwu Qian, Pengcheng Gao, Xiao-Yong Wei, Yaowei Wang, Yonghong Tian, and Wen Gao. 2023a. Large-scale multi-modal pre-trained models: A comprehensive survey. *Machine Intelligence Research*, 20(4):447–482.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023b. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Colin Wei, Sang Michael Xie, and Tengyu Ma. 2021a. Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning. *Advances in Neural Information Processing Systems*, 34:16158–16170.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021b. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Chenwei Wu, Holden Lee, and Rong Ge. 2023. Connecting pre-trained language model and downstream task via properties of representation. *Advances in Neural Information Processing Systems*, 36:47216–47238.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhensu Chen, Xiaopeng Zhang, and Qi Tian. 2023. Qa-lora: Quantization-aware low-rank adaptation of large language models. *arXiv preprint arXiv:2309.14717*.

Shanglong Yang, Zhipeng Yuan, Shunbao Li, Ruoling Peng, Kang Liu, and Po Yang. 2024. Gpt-4 as evaluator: Evaluating large language models on pest management in agriculture. *arXiv preprint arXiv:2403.11858*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.