UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

MaCP: Minimal yet Mighty Adaptation via Hierarchical Cosine Projection

Shen, Y.; Bi, Q.; Huang, Jia-Hong; Zhu, Hongyi; Pimentel, A.D.; Pathania, A.

DOI
10.18653/v1/2025.acl-long.1006

Publication date
2025

Document Version
Final published version

Published in
The 63rd Annual Meeting of the Association for Computational Linguistics (ACL 2025) : proceedings of the conference

License
CC BY

Link to publication

Citation for published version (APA):
Shen, Y., Bi, Q., Huang, J.-H., Zhu, H., Pimentel, A. D., & Pathania, A. (2025). MaCP: Minimal yet Mighty Adaptation via Hierarchical Cosine Projection. In W. Che, J. Nabende, E. Shutova, & M. T. Pilehvar (Eds.), *The 63rd Annual Meeting of the Association for Computational Linguistics (ACL 2025) : proceedings of the conference: ACL 2025 : July 27-August 1, 2025* (Vol. 1, pp. 20602–20618). Association for Computational Linguistics. https://doi.org/10.18653/v1/2025.acl-long.1006

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:17 Oct 2026

MaCP: Minimal yet Mighty Adaptation via Hierarchical Cosine Projection

Yixian Shen, Qi Bi, Jia-Hong Huang, Hongyi Zhu, Andy D. Pimentel, Anuj Pathania
University of Amsterdam, Amsterdam, the Netherlands

{y.shen, q.bi, j.huang, h.zhu, a.d.pimentel, a.pathania}@uva.nl

Abstract

We present a new adaptation method MaCP, Minimal yet Mighty adaptive Cosine Projection, that achieves exceptional performance while requiring minimal parameters and memory for fine-tuning large foundation models. Its general idea is to exploit the superior energy compaction and decorrelation properties of cosine projection to improve both model efficiency and accuracy. Specifically, it projects the weight change from the low-rank adaptation into the discrete cosine space. Then, the weight change is partitioned over different levels of the discrete cosine spectrum, and each partition's most critical frequency components are selected. Extensive experiments demonstrate the effectiveness of MaCP across a wide range of single-modality tasks, including natural language understanding, natural language generation, text summarization, as well as multi-modality tasks such as image classification and video understanding. MaCP consistently delivers superior accuracy, significantly reduced computational complexity, and lower memory requirements compared to existing alternatives.

1 Introduction

Large language models (LLMs) have demonstrated exceptional performance across a wide range of natural language processing (NLP) tasks, owing to their strong generalization ability achieved through training on large-scale corpora (Touvron et al., 2023; Liu et al., 2023). However, their zero-shot performance on many downstream tasks often remains suboptimal, failing to meet task-specific requirements. A common approach to address this limitation is full fine-tuning, where the model is adapted to a specific downstream task. However, the immense parameter size of LLMs makes this process computationally expensive and resource-intensive. For example, fine-tuning the state-of-the-art LLaMA 3.1 (Vavekanand and Sam, 2024) model with 70 billion parameters requires nearly 500GB of GPU memory, making it impractical for many real-world deployments.

Table 1: Comparison of methods across memory optimization and computation intensity GFLOPs.

Method	**Memory**			**GFLOPs**
	Optimizer	**Gradient**	**Activation**	
Full FT	✗	✗	✗	Highest
LoRA	✓	✓	✗	High
AdaLoRA	✓	✓	✗	High
DoRA	✓	✓	✓	High
VeRA	✓	✓✓	✓	Low
AFLoRA	✓	✓✓	✓	Medium
LaMDA	✓	✓✓	✓✓	Low
FourierFT	✓	✓✓	✓✓	Low
MaCP (Ours)	✓✓	✓✓	✓✓	**Minimal**

Parameter-efficient fine-tuning (PEFT) methods effectively address these challenges. Key PEFT approaches include low-rank adaptation (LoRA) (Hu et al., 2022), adapter tuning (He et al., 2021), and prompt tuning (Lester et al., 2021). LoRA and its variants (Liu et al., 2024) reduce trainable parameters by decomposing weight updates into two low-rank matrices, achieving SOTA accuracy. However, this efficiency in parameter count does not directly translate to reduced memory or compute cost. LoRA, for example, expands the effective embedding dimensionality, increasing FLOPs and requiring storage of high-dimensional activations and optimizer states during backpropagation (Azizi et al., 2024). As summarized in Table 1, existing PEFT techniques still suffer from partial inefficiencies, while methods like VeRA and LaMDA mitigate some memory overhead by optimizing specific components (e.g., gradients or optimizer states), they continue to incur significant costs in activation storage, overall memory footprint, and computational intensity.

Recent research has begun exploring the frequency domain to compress and optimize model weights while preserving expressive capability. For instance, Gao et al.(Gao et al., 2024) demonstrated that applying the Discrete Fourier Transform (DFT) to large language model (LLM) weight matrices can significantly reduce trainable parameters by focusing on the most informative spectral components, highlighting the promise of frequency-

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 20602–20618
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

domain techniques for efficient fine-tuning. However, DFT is inherently suited for periodic signals(Nagai et al., 2020), while the long-range dependencies in language context are often diverse and non-periodic (Sun et al., 2021; Wen et al., 2023). Moreover, DFT operates in the complex domain, introducing computational overhead and potential numerical instability (Press, 2007), particularly for large-scale models (Gao et al., 2024). These numerical inaccuracies can degrade performance, making the energy compaction and stable adaptation of LLMs in the frequency domain less effective than intended.

In this paper, we propose a novel fine-tuning approach that leverages the Discrete Cosine Transform (DCT) to address the existing bottlenecks. DCT is particularly well-suited for non-periodic signals and exhibits superior energy compaction and decorrelation properties, enabling a more efficient representation of critical information with fewer coefficients. To further enhance this efficiency, we partition the frequency domain using the discrete cosine coefficient matrix to identify and select the most informative components within the context. Specifically, MaCP simplifies the computational process compared to DFT-based approaches while preserving the expressive capacity of LLMs.

Building on these insights, by systematically selecting the most informative frequency components, compressing the weight updates, and then applying the inverse DCT (iDCT) to revert to the time domain, the proposed MaCP not only preserves model expressiveness but also drastically reduces memory and compute requirements. As illustrated in Fig.1, MaCP minimizes the footprint of activations, gradients, and optimizer states, achieving significantly lower GPU memory usage than existing methods. By eliminating redundancy and reducing overhead across the training pipeline, MaCP offers a scalable and resource-efficient solution for fine-tuning large language models.

Our contribution can be summarized as follows.

- We propose a novel Hierarchical Cosine Projection (MaCP) method for large language models, enabling minimal resource adaptation while preserving performance.
- We introduce an innovative frequency partitioning and selection strategy that exploits the Discrete Cosine Transform's (DCT) energy compaction property to enhance computational efficiency.
- We conduct extensive experiments on diverse benchmarks, demonstrating that MaCP outperforms existing PEFT methods in both performance and resource efficiency.

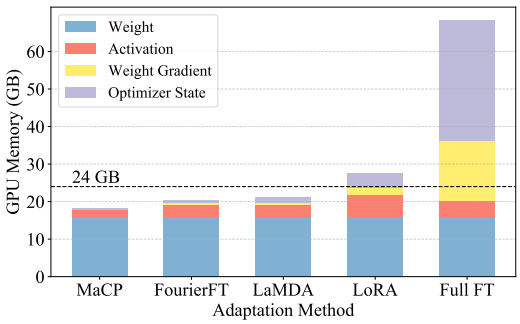

Figure 1: GPU memory usage of LLaMA3.1-8B on different fine-tuning methods

2 Related Work

Low-Rank Adaptation (LoRA) (Hu et al., 2022) reduces trainable parameters by learning low-rank matrices, bypassing full-weight updates and minimizing memory usage for gradients and optimizers. Unlike adapter-based methods (He et al., 2021; Pfeiffer et al., 2020; Lin et al., 2020; Liao et al., 2023b,a), LoRA incurs no inference overhead as low-rank updates merge with frozen weights. However, scaling LoRA to larger models and complex tasks remains challenging. Recent methods, including AdaLoRA (Zhang et al., 2023), VeRA (Kopiczko et al., 2023), QLoRA (Dettmers et al., 2024), DoRA (Liu et al., 2024), and LaMDA (Azizi et al., 2024), improve parameter allocation and decomposition but still face scalability issues on large models.

Frequency-based Spectrum Learning has been explored to reduce trainable parameters while maintaining model capacity. Prior works (Xu et al., 2020; Tang et al., 2022; Yang and Xie, 2016) demonstrated the effectiveness of compact and sparse spectral representations. Gao et al. (Gao et al., 2024) applied the Fourier Transform to fine-tune a subset of spectral coefficients, showcasing the potential of sparse spectrum adaptation in large foundation models. However, DFT adds complex operations, and the asymmetry between the DFT and its inverse increases computational overhead.

Discrete Cosine Transform has also been effectively applied in various deep learning contexts, demonstrating its potential for model compression and efficiency improvements. For instance, Xu et al. analyzed the spectral bias from a frequency perspective and proposed a learning-based frequency selection method to identify and remove trivial frequency components without loss of accuracy (Xu

Algorithm 1 MaCP Algorithm

Input: Input tensor x, number of parameters n, scaling factor α, input dimension d_1, output dimension d_2, energy ratio δ, pre-trained layer weights W
Output: Transformed tensor h

Initialization:
$W_F = \text{DCT}(base_layer.weight)$ //DCT for weights, Eq(1)
// Partition frequencies into low, mid, and high components.
$\mathcal{M}_{\text{low}}, \mathcal{M}_{\text{mid}}, \mathcal{M}_{\text{high}} \leftarrow \text{DistancePartition}(W_F)$ Eq(5,6,7)

// Select top-$(n_{\mathcal{M}} \times \delta)$ frequencies by energy
$n_{\mathcal{M}} \times \delta \leftarrow \text{RankTopEnergyFreq}(\mathcal{M})$
// Randomly select the rest of frequencies
$n_{\mathcal{M}} \times (1-\delta) \leftarrow \text{RandomSelectFreq}(\mathcal{M})$
// Stratified sampling from $\mathcal{M}_{\text{low}}$, $\mathcal{M}_{\text{mid}}$, $\mathcal{M}_{\text{high}}$, Eq(8)
$n \leftarrow n_{\mathcal{M}_{\text{low}}} + n_{\mathcal{M}_{\text{mid}}} + n_{\mathcal{M}_{\text{high}}}$
// Initialize spectral coefficients
$\Delta W_F \leftarrow \text{KaimingInitial}()$
Forward Pass:
// Set n selected freq. trainable and froze $(d_1 \times d_2 - n)$ freq.
Set n frequencies $\leftarrow$ requireGrad(True)
// Compute ΔW_T using inverse DCT based on Eq(9,10)
$\Delta W_T \leftarrow \text{iDCT}(\Delta W_F) \times \alpha$
// Merge ΔW with base layer output
$h \leftarrow W + \Delta W_T$
return h

et al., 2020). Ehrlich and Davis (2019). leveraged DCT within the context of JPEG-compressed images, redefining convolution and batch normalization with a tunable approximation for ReLU operations (Ehrlich and Davis, 2019). Furthermore, Zhu demonstrated that DCT could effectively recover data with very few parameters, highlighting its potential for model compression and efficiency (Zhu et al., 2024; Cheng et al., 2024).

3 Methodology

Fig. 2 gives an overview of the proposed Selective Discrete Cosine Transform fine-tuning (MaCP) method. Different from the vanilla LoRA (low-rank adaptation) paradigm, MaCP operates in the frequency domain by learning a set of spectral coefficients on the cosine basis instead of directly modifying the spatial weights. The proposed MaCP identifies and retains the most critical frequency components by applying the DCT to the pre-trained weights. Then, it selects some spectral coefficients to learn trainable parameters, which form the spectral matrix. Finally, the iDCT is applied to the modified spectral matrix to transform it back to the spatial domain so as to update the weight changes, as detailed in Algorithm1.

3.1 Cosine Projection Using DCT

The first step is to transform the weight matrices of the LLM into the frequency domain using the

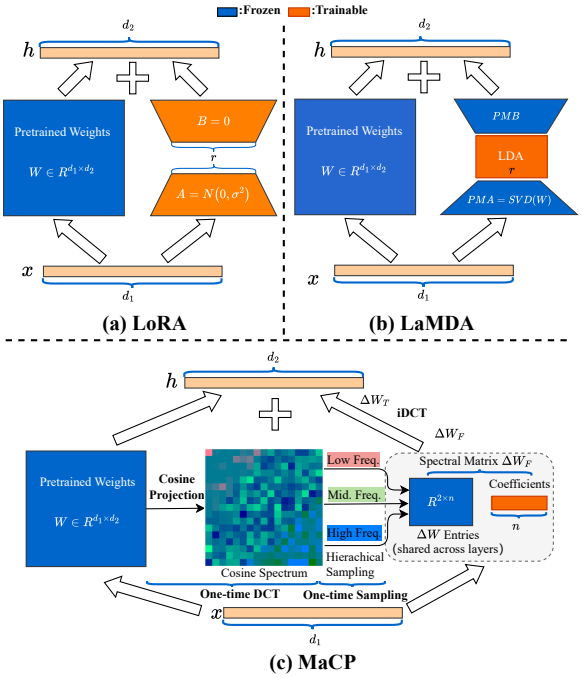

Figure 2: (a) LoRA (Hu et al., 2022) reduces trainable parameters by decomposing weight updates into low-rank matrices. (b) LaMDA (Azizi et al., 2024) further reduces trainable parameters through low-dimensional adaptation (LDA). (c) MaCP (ours) applies the Discrete Cosine Transform (DCT) to pre-trained weights, partitioning the frequency domain into low, middle, and high frequencies. A hierarchical sampling strategy then selects the most critical spectral coefficients for fine-tuning, reducing trainable parameters to n per layer, which is significantly smaller compared to LoRA and LaMDA.

Discrete Cosine Transform (DCT). DCT is particularly advantageous due to its strong energy compaction properties, where the majority of the signal's energy is concentrated in a small number of low-frequency components. Specifically, given a weight matrix $W[i,j]$ of size $M \times N$, the 2D DCT is mathematically defined as:

$$W_F[u,v] = \alpha(u)\alpha(v) \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} W[i,j] \times \cos\left[\frac{\pi}{M}\left(i+\frac{1}{2}\right)u\right] \cos\left[\frac{\pi}{N}\left(j+\frac{1}{2}\right)v\right], \quad (1)$$

where $u = 0, 1, \ldots, M-1$ and $v = 0, 1, \ldots, N-1$ represent the frequency indices. The scaling factors $\alpha(u)$ and $\alpha(v)$ are defined as:

$$\alpha(x) = \begin{cases} \sqrt{\frac{1}{M}}, & \text{if } x = 0 \\ \sqrt{\frac{2}{M}}, & \text{if } x = 1, 2, \ldots, M-1 \end{cases} \quad (2)$$

This transformation converts the spatial (or time-domain) information of the weight matrix into a frequency-domain representation, where each element $W_F[u,v]$ corresponds to a specific frequency

component of the original matrix. Low-frequency components (where u and v are small) typically contain the most significant information, making them prime candidates for focused fine-tuning and significantly reducing the number of parameters that need to be fine-tuned.

3.2 Hierarchical Cosine Spectrum

The second step systematically partitions the frequency spectrum into three groups of distinct regions, namely, low, mid, and high frequencies. Each of these regions captures different aspects of the model's behavior, with low frequencies generally representing more global structures and high frequencies capturing finer.

Distance-Based Frequency Partitioning. We define the frequency domain as a 2D grid where each point (u, v) corresponds to a specific frequency component in the Discrete Cosine Transform (DCT) of the weight matrix. The distance of each point from the origin (which represents the lowest frequency) is given by:

$$d(u, v) = \sqrt{u^2 + v^2} \tag{3}$$

where u and v are the frequency indices corresponding to the horizontal and vertical directions in the DCT matrix, respectively. The maximum distance from the origin is:

$$d_{\text{max}} = \sqrt{\left(\frac{M}{2}\right)^2 + \left(\frac{N}{2}\right)^2} \tag{4}$$

where M and N are the dimensions of the DCT matrix. The frequency domain is partitioned into three distinct regions: low, medium, and high frequencies. These regions are defined based on their distance from the origin:

Low-Frequency Components ($\mathcal{M}_{\text{low}}$): Low frequencies are located close to the origin and are defined as:

$$\mathcal{M}_{\text{low}} = \{(u, v) : d(u, v) \leq \frac{d_{\text{max}}}{3}\} \tag{5}$$

These components capture broad, global patterns and contain the majority of the energy due to the DCT's energy compaction property.

Medium-Frequency Components ($\mathcal{M}_{\text{mid}}$): Medium frequencies lie at an intermediate distance from the origin and are defined as:

$$\mathcal{M}_{\text{mid}} = \{(u, v) : \frac{d_{\text{max}}}{3} < d(u, v) \leq \frac{2 \cdot d_{\text{max}}}{3}\} \tag{6}$$

These components capture finer details and are essential for representing medium-scale structures.

High-Frequency Components ($\mathcal{M}_{\text{high}}$): High frequencies are located farthest from the origin and are defined as:

$$\mathcal{M}_{\text{high}} = \{(u, v) : d(u, v) > \frac{2 \cdot d_{\text{max}}}{3}\} \tag{7}$$

These components capture detailed features, which may include noise, but they are important for tasks requiring fine resolution.

Within each partition ($\mathcal{M}_{\text{low}}$, $\mathcal{M}_{\text{mid}}$, $\mathcal{M}_{\text{high}}$), we apply a hybrid selection strategy that combines energy-based selection with a diversity-enhancing mechanism, where the top $n_{\mathcal{M}_k} \times \delta$ coefficients are first selected based on energy (with δ defaulting to 0.7 unless otherwise stated), followed by random selection for additional coefficients and stratified sampling to balance high-energy components and diverse frequencies across all partitions, in Algorithm 1 in Appendix. For more details on δ, see the ablation study in section 4. The set of selected frequencies across all partitions is given by:

$$n = \bigcup_{k=1}^{p} n_{\mathcal{M}_k}, \tag{8}$$

where $n_{\mathcal{M}_k}$ is the set of selected frequencies from partition $\mathcal{M}_k$ and $n_{\mathcal{M}_k} = n_{\text{energy},\mathcal{M}_k} + n_{\text{random},\mathcal{M}_k}$.

3.3 Cosine Spectrum Learning via iDCT

Once the most informative frequency components have been selected, the final step is to update the weights in the frequency domain and then transform them back to the space domain using the inverse Discrete Cosine Transform (iDCT). The updated weight matrix ΔW is obtained via:

$$\Delta W_T = \text{iDCT}(\Delta W_F), \tag{9}$$

where ΔW_F represents the modified frequency-domain weights, and iDCT is mathematically defined as:

$$\begin{aligned} W[i, j] = &\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v) W_F[u, v] \\ &\times \cos\left[\frac{\pi}{M}\left(i + \frac{1}{2}\right)u\right] \cos\left[\frac{\pi}{N}\left(j + \frac{1}{2}\right)v\right]. \end{aligned} \tag{10}$$

Note that only the selected frequency components in ΔW_F are updated. The transformation back to the spatial domain integrates this fine-tuned information, preparing the model for inference or further training.

3.4 Memory Efficiency Analysis

The activation memory footprint of MaCP is significantly reduced because only a small subset of spectral coefficients n is updated while keeping the remaining parameters frozen. The activation memory for MaCP consists of two components: (1) memory for the forward pass, requiring $B \cdot S \cdot H$ to store activations for all input-output pairs, where S is the sequence length, H is the embedding dimension, and B is the batch size; and (2)memory for backpropagation, limited to the selected spectral coefficients, $B \cdot n$. Thus, the total activation memory footprint for MaCP is:

$$M_{act}^{MaCP} = B \cdot S \cdot H + B \cdot n, \tag{11}$$

compared to LoRA, which requires:

$$M_{act}^{LoRA} = B \cdot S \cdot H + B \cdot S \cdot H, \tag{12}$$

where $n < S \cdot H$, ensuring MaCP consumes less memory. For example, in a layer with $S = 2048$, $H = 4096$, $B = 1$, $n = 1000$, and $r = 32$ in LLaMA3.1-8B, reduces the activation memory footprint by 50.01% compared to LoRA, which underscore MaCP's efficiency, particularly in memory-constrained environments.

4 Experiments

4.1 Baselines

The proposed MaCP is compared with the following baselines: **Full Fine-Tuning (FF)** updates all parameters of the model. **Adapter Tuning** (Houlsby et al., 2019; Lin et al., 2020; Rücklé et al., 2020; Pfeiffer et al., 2020) introduces adapter layers between the self-attention and MLP modules. **LoRA** (Hu et al., 2022) approximates weight updates using low-rank matrices to minimize trainable parameters, while **AdaLoRA** (Zhang et al., 2023) dynamically allocates the parameter budget based on importance scores. **DoRA** (Liu et al., 2024) decomposes pre-trained weights into magnitude and direction, applying LoRA for directional updates. **VeRA** (Kopiczko et al., 2023) uses a single pair of low-rank matrices across all layers with small scaling vectors to reduce parameters while maintaining performance. **FourierFT** (Gao et al., 2024) fine-tunes a subset of spectral coefficients in the Fourier domain, and **LaMDA** (Azizi et al., 2024) employs low-dimensional adaptation to further reduce trainable parameters.

4.2 Natural Language Understanding

Models, Datasets, and Implementation. We evaluate MaCP on the GLUE benchmark (Wang et al., 2019) using RoBERTa (Liu et al., 2019) in both Base and Large configurations, covering diverse NLU tasks for comprehensive assessment. **Implementation Details.** MaCP uses 700 out of 768^2 spectral coefficients for RoBERTa Base and 1024^2 for RoBERTa Large, selecting the most critical components uniformly across all layers. To ensure fair comparison, we adopt the same experimental settings as LoRA, with hyperparameters provided in the appendix.

Results and Analysis The results are presented in Tab. 2, where we report the median performance across 5 random seed runs, with the best epoch selected for each run. The proposed MaCP consistently delivers superior or comparable performance to baseline methods while significantly reducing the number of trainable parameters. For instance, MaCP achieved an 80.7% accuracy on RTE and a 92.0 Pearson correlation on STS-B, outperforming methods like LoRA and FourierFT with significantly fewer parameters. On SST-2, MaCP maintained a high accuracy of 96.2%, matching or slightly outperforming other approaches. Additionally, it recorded a 94.5% accuracy on QNLI, further highlighting its effectiveness. Across all tasks, MaCP shows robust generalization while requiring minimal training parameters.

4.3 Natural Language Generation

Models and Datasets. We evaluate MaCP on the E2E natural language generation (NLG) task (Novikova et al., 2017) by fine-tuning GPT-2 (Medium and Large) (Radford et al., 2019) models, which are decoder-only architectures with 24 and 36 transformer blocks.

Implementation Details. We fine-tune LoRA, DoRA, FourierFT, VeRA, and the proposed MaCP method on the GPT-2 Medium and Large models using a linear learning rate scheduler over 5 epochs. Results are averaged over 3 runs, and detailed hyperparameters are in the Appendix.

Results and Analysis. Tab. 3 shows that MaCP consistently outperforms other methods across most metrics while requiring significantly fewer trainable parameters. Specifically, MaCP reduces trainable parameters by 9.59% and 6.6% compared to LoRA for GPT-2 Medium and Large models, respectively, and achieves a 30% parameter reduction compared to FourierFT.

4.4 Text Summarization

Models and Datasets. We assess MaCP's effectiveness on text summarization using the BART-Large model (Lewis et al., 2019), evaluated on two stan-

	Model & Method	# Trainable Parameters	MaCP's Para. Savings	SST-2↑ (Acc.)	MRPC↑ (Acc.)	CoLA↑ (MCC)	QNLI↑ (Acc.)	RTE↑ (Acc.)	STS-B↑ (PCC)	Avg.
BASE	FF	125M	**99.99%**	94.8	90.2	63.6	92.8	78.7	91.2	85.22
	BitFit	0.1M	**83.00%**	93.7	**92.7**	62.0	91.8	**81.5**	90.8	**85.42**
	Adpt^{D}	0.9M	**98.11%**	94.7±0.3	88.4±0.1	62.6±0.9	93.0±0.2	75.9±2.2	90.3±0.1	84.15
	LoRA	0.3M	**94.33%**	**95.1**±0.2	89.7±0.7	63.4±1.2	**93.3**±0.3	78.4±0.8	**91.5**±0.2	85.23
	AdaLoRA	0.3M	**94.33%**	94.5±0.2	88.7±0.5	62.0±0.6	93.1±0.2	81.0±0.6	90.5±0.2	84.97
	DoRA	0.3M	**94.33%**	94.9±0.5	89.9±0.5	63.7±1.1	**93.3**±0.2	78.9±0.6	**91.5**±0.2	85.37
	VeRA	0.043M	**60.47%**	94.6±0.1	89.5±0.5	**65.6**±0.9	91.8±0.3	78.7±0.7	90.7±0.2	85.15
	LaMDA	0.089M	**80.90%**	94.5±0.2	89.3±0.3	**65.6**±0.5	91.2±0.6	78.9±0.3	90.5±0.3	85.01
	FourierFT	0.024M	**29.17%**	94.2±0.3	90.0±0.8	63.8±1.6	92.2±0.1	79.1±0.5	90.8±0.2	85.02
	MaCP	**0.017M**	-	94.2±0.2	89.7±0.7	64.6±1.4	92.4±0.1	80.7±0.9	90.9±0.3	**85.42**
LARGE	FF	356M	**99.99%**	96.3	90.9	68.0	94.7	86.6	92.4	88.11
	Adpt^{P}	3M	**98.87%**	96.1±0.3	90.2±0.7	**68.3**±1.0	94.7±0.2	83.8±2.9	92.1±0.7	87.55
	Adpt^{P}	0.8M	**95.75%**	**96.6**±0.2	89.7±1.2	67.8±2.5	94.7±0.3	80.1±2.9	91.9±0.4	86.82
	Adpt^{H}	6M	**99.43%**	96.2±0.3	88.7±2.9	66.5±4.4	94.7±0.2	83.4±1.1	91.0±1.7	86.75
	Adpt^{H}	0.8M	**95.75%**	96.3±0.5	87.7±1.7	66.3±2.0	94.7±0.2	72.9±2.9	91.5±0.5	84.90
	LoRA	0.8M	**95.75%**	96.2±0.5	90.2±1.0	68.2±1.9	**94.8**±0.3	85.2±1.1	92.3±0.5	87.82
	DoRA	0.9M	**96.22%**	96.4±0.4	**91.0**±1.1	67.2±1.6	**94.8**±0.3	85.4±1.1	92.1±0.3	87.82
	VeRA	0.061M	**44.26%**	96.1±0.1	90.9±0.7	68.0±1.8	94.4±0.2	85.9±0.7	91.7±0.8	87.83
	LaMDA	0.12M	**71.67%**	96.0±0.1	90.9±0.2	67.1±1.2	94.2±0.5	85.3±0.5	91.3±0.4	87.47
	FourierFT	0.048M	**29.17%**	96.0±0.2	90.9±0.3	67.1±1.4	94.4±0.4	**87.4**±1.6	91.9±0.4	87.95
	MaCP	**0.034M**	-	96.2±0.3	90.9±0.3	67.7±1.4	94.5±0.5	**87.4**±1.1	**92.0**±0.5	**88.12**

Table 2: Performance of various fine-tuning methods on 6 GLUE tasks, using base and large models. Metrics include MCC for CoLA, PCC for STS-B, and accuracy for other tasks. Results are medians of 5 runs with different seeds; the best scores in each category are bolded.

	Method	# Tr. Para.	BLEU	NIST	METE.	ROU-L	CIDEr
GPT-2 Medium	FT^{1}	354.92M	68.2	8.62	46.2	71.0	2.47
	$\text{Adpt}^{\text{L}^{1}}$	0.37M	66.3	8.41	45.0	69.8	2.40
	$\text{Adpt}^{\text{L}^{1}}$	11.09M	68.9	8.71	46.1	71.3	2.47
	$\text{Adpt}^{\text{H}^{1}}$	11.09M	67.3	8.50	46.0	70.7	2.44
	LoRA	0.35M	68.9	8.76	46.6	71.5	2.51
	DoRA	0.36M	69.2	8.79	46.9	71.7	2.52
	VeRA	0.35M	**70.1**	8.81	46.6	71.5	2.50
	FourierFT	0.048M	69.1	**8.82**	47.0	71.8	2.51
	MaCP	**0.034M**	**70.1**	**8.82**	**47.2**	**71.9**	**2.53**
GPT-2 Large	FT^{1}	774.03M	68.5	8.78	46.0	69.9	2.45
	$\text{Adpt}^{\text{L}^{1}}$	0.88M	69.1	8.68	46.1	71.0	2.49
	$\text{Adpt}^{\text{L}^{1}}$	23.00M	68.9	8.70	46.1	71.3	2.45
	LoRA	0.77M	69.4	8.81	46.5	**71.9**	2.50
	DoRA	0.79M	69.8	8.83	46.9	**71.9**	2.50
	VeRA	0.17M	**70.3**	8.85	46.6	71.6	2.54
	FourierFT	0.072M	70.2	8.90	47.0	71.8	2.50
	MaCP	**0.051M**	**70.3**	**8.92**	**47.1**	**71.9**	**2.55**

Table 3: Performance comparison of different fine-tuning methods on GPT-2 Medium and GPT-2 Large models, evaluated using BLEU, NIST, METEOR, ROUGE-L, and CIDEr metrics. [1] denotes results sourced from previous studies. The best score for each metric is highlighted in bold.

Model	Method	# Tr. Para.	MT-Bench	Vicuna
LLaMA2-7B	LoRA	159.9M	5.19	6.85
	DoRA	163.7M	5.20	6.86
	VeRA	1.6M	5.18	6.82
	FourierFT	0.064M	5.09	6.85
	MaCP	**0.045M**	**5.22**	**6.87**
LLaMA2-13B	LoRA	250.3M	5.77	7.38
	DoRA	264.5M	5.79	7.47
	VeRA	2.4M	**5.93**	7.41
	FourierFT	0.08M	5.82	7.49
	MaCP	**0.056M**	**5.93**	**7.55**
LLaMA3.1-8B	LoRA	183.3M	5.65	7.11
	DoRA	186.9M	5.66	**7.19**
	VeRA	1.9M	5.61	7.13
	FourierFT	0.064M	5.67	7.15
	MaCP	**0.05M**	**5.69**	**7.19**

Table 4: The average scores on MT-Bench and Vicuna assessed by GPT-4. The higher score is better.

dard benchmarks: XSUM (Narayan et al., 2018) and CNN/DailyMail (Hermann et al., 2015).

Implementation Details. We compare MaCP with SOTA baselines under consistent settings. For LaMDA, we use a rank of $r = 32$, and for MaCP, we select $n = 5000$ cosine spectrum coefficients. All models are trained with a learning rate of 2×10^{-4}; batch sizes are 32 for XSUM and 64 for CNN/DailyMail. Training runs for 25 epochs on XSUM and 15 on CNN/DailyMail.

Results and Analysis. Table 5 shows that MaCP achieves the best overall performance on both XSUM and CNN/DailyMail, despite using drastically fewer trainable parameters. On XSUM, MaCP outperforms all baselines with ROUGE-1/2/L scores of 45.21/22.19/37.10, even surpassing full fine-tuning, which requires over 2400× more parameters. Similarly, on CNN/DailyMail, MaCP delivers top results with 45.09/21.97/42.29, outperforming all other PEFT baselines. These results demonstrate that MaCP preserves key semantic and structural information during fine-tuning, while operating under extreme parameter efficiency.

Notably, MaCP's advantage over FourierFT underscores the benefit of using cosine over Fourier bases. DCT's real-valued, non-periodic decomposition aligns better with the structure of language data, enabling more effective compression and adaptation. Compared to LaMDA and VeRA, which also aim for compactness, MaCP delivers significantly better ROUGE scores while using fewer parameters.

Method	#Param.	XSUM	CNN/DailyMail
Full fine-tuning	415M	45.14/22.27/37.25	44.16/21.28/40.90
LoRA	8.6M	43.95/20.72/35.68	45.03/21.84/42.15
DoRA	8.8M	44.23/21.17/36.02	44.92/21.89/42.09
VeRA	0.92M	43.72/20.33/35.11	44.87/21.26/40.25
LaMDA	0.85M	43.92/20.68/35.21	44.12/21.16/40.45
Fourier	0.33M	44.39/20.91/36.05	44.42/21.07/40.53
MaCP	**0.17M**	**45.21/22.19/37.10**	**45.09/21.97/42.29**

Table 5: Comparison of fine-tuning methods on XSUM and CNN/DailyMail benchmarks. Bold highlights the best results.

4.5 Instruction Tuning

Models, Datasets, and Implementation. We fine-tune LLaMA2-7B, LLaMA2-13B, and LLaMA3.1-8 using MaCP and baseline methods on the Alpaca dataset (Taori et al., 2023). For evaluation, responses are generated for pre-defined questions from MT-Bench (Zheng et al., 2024) and Vicuna Eval datasets, and scored by GPT-4 on a 10-point scale. Following prior work (Dettmers et al., 2024, 2022), LoRA, DoRA, and VeRA are applied to all linear layers except the top one, while FourierFT follows the settings in (Gao et al., 2024). For MaCP, we set $n = 700$. To train on a single GPU, we adopt the QLoRA quantization technique (Dettmers et al., 2024). All methods are trained for one epoch, and the average scores of all responses are reported. Hyperparameter details are provided in the Appendix.

Results and Analysis. Tab. 4 show that the MaCP consistently outperforms or matches baseline methods across different LLaMA models. It achieves the highest scores on both MT-Bench and Vicuna for LLaMA2-7B and LLaMA2-13B, demonstrating its effectiveness in generating high-quality responses with significantly fewer trainable parameters. For LLaMA2-7B, MaCP requires only 0.045M parameters, reducing 99.7% compared to LoRA, yet it delivers superior performance. Similarly, for LLaMA2-13B, MaCP matches the top score of 5.93 on MT-Bench and achieves the highest score of 7.55 on Vicuna, with only 0.056M parameters. For LLaMA3.1-8B, MaCP shows competitive results, outperforming other methods with a score of 5.69 on MT-Bench and matching the top score of 7.19 on Vicuna, with minimal trainable parameters.

4.6 Image Classification

Models, Datasets, and Implementation. We evaluate our method on the Vision Transformer (ViT) (Dosovitskiy et al., 2020) in Base and Large variants using CIFAR-100 (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), and OxfordPets (Parkhi et al., 2012). LoRA, DoRA, VeRA, and six other baselines are applied to the query and value layers of ViT, while training only the classification head is denoted as "Head." We set $r = 16$ for LoRA, $n = 3000$ for FourierFT, and $n = 2400$ for MaCP, with learning rates and weight decay tuned for all methods. Training is capped at 10 epochs, with hyperparameter details provided in the Appendix.

Model	Method	# Train. Para.	CIFAR100	DTD	EuroSAT	OxfordPets
ViT-B	Head	-	84.3	69.8	88.7	90.3
	Full	85.8M	**92.4**	**77.7**	**99.1**	**93.4**
	LoRA	581K	92.1	75.2	98.4	93.2
	DoRA	594K	92.3	75.3	98.7	93.2
	VeRA	57.3K	91.7	74.6	98.5	**93.4**
	FourierFT	72K	91.2	75.1	98.8	93.2
	MaCP	**50.4K**	91.7	75.9	**99.1**	**93.4**
ViT-L	Head	-	84.7	73.3	92.6	91.1
	Full	303.3M	93.6	81.8	**99.1**	94.4
	LoRA	1.57M	94.9	81.8	98.63	**94.8**
	DoRA	1.62M	**95.1**	81.8	98.8	**94.8**
	VeRA	130.5K	94.2	81.6	98.6	93.7
	FourierFT	144K	93.7	81.2	98.7	94.5
	MaCP	**100.9K**	94.5	**81.9**	99.0	**94.8**

Table 6: Performance of various fine-tuning methods on ViT-B and ViT-L models across different datasets. The best results for each dataset are highlighted in bold.

Results and Analysis. Tab. 6 presents the performance on ViT-B and ViT-L across four image classification datasets. For the ViT-B model, MaCP demonstrates competitive performance with only 50.4K trainable parameters, while LoRA and DoRA require over 10 times more parameters. Specifically, MaCP matches the full fine-tuning performance on EuroSAT and OxfordPets, achieving 99.1% and 93.4% accuracy, respectively. For the ViT-L model, MaCP achieves near-optimal performance with only 100.9K parameters. MaCP records the best result on DTD at 81.9%, while matching the top accuracy on OxfordPets at 94.8%.

Model	Method	# Train. Para.	TVQA	How2QA	TVC	YC2C
VL-BART	Full	228.9M	**76.3**	73.9	45.7	**154**
	LoRA	11.8M	75.5	72.9	44.6	140.9
	DoRA	11.9M	**76.3**	74.1	45.8	145.4
	VeRA	1.3M	75.9	73.8	44.7	142.6
	FourierFT	1.5M	76.2	73.1	45.5	147.3
	MaCP	**1.1M**	**76.3**	**74.3**	**45.9**	**154**

Table 7: Multi-task evaluation results on TVQA, How2QA, TVC, and YC2C using the VL-BART backbone. MaCP achieves top or near-top scores across all tasks with significantly fewer trainable parameters, demonstrating its efficiency and effectiveness.

4.7 Video-Text Understanding

Models, Datasets, and Implementation. We compare DoRA, LoRA, and full fine-tuning on VL-BART, which integrates CLIP-ResNet101 (Radford et al., 2021) as the vision encoder and BART-Base (Lewis et al., 2019) as the language model. The evaluation covers four tasks: TVQA (Lei et al., 2018) and How2QA (Li et al., 2020) for video question answering, and TVC (Lei et al., 2020) and YC2C (Zhou et al., 2018) for video captioning. Fol-

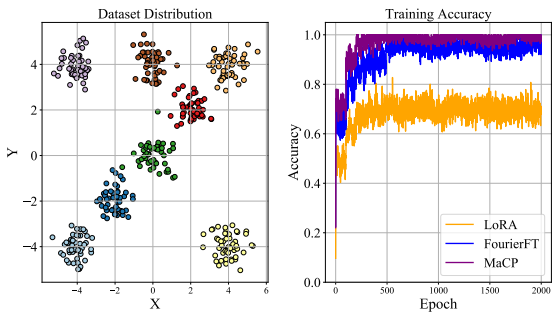

Figure 3: Left: Dataset distribution for the synthetic classification task, where each color represents a different class. Right: Training accuracy over 2000 epochs for LoRA, FourierFT, and MaCP. Although all methods use the same number of trainable parameters, LoRA exhibits significant oscillations and fails to converge to 100% accuracy, indicating performance bottlenecks. FourierFT quickly reaches near-perfect accuracy within 500 epochs with stable performance. Our proposed MaCP approach outperforms both methods by converging faster (around 450 epochs) and maintaining higher accuracy with minimal fluctuations.

lowing (Sung et al., 2022), VL-BART is fine-tuned in a multi-task setup. We set $r = 128$ for LoRA, $n = 6000$ for FourierFT, and $n = 4500$ for MaCP, with tuned learning rates and weight decay, training for 7 epochs.

Results and Analysis. Table 7 presents the multi-task evaluation results for video-text tasks using the VL-BART backbone. MaCP achieves top or near-top performance across all tasks while using significantly fewer trainable parameters. For TVQA and YC2C, MaCP matches the highest scores of **76.3** and **154**, respectively, while reducing parameters by over 99% compared to full fine-tuning. On How2QA, MaCP records the highest score of **74.3**, outperforming all baselines. For TVC, MaCP achieves a strong result of **45.9**, slightly surpassing DoRA and FourierFT, further demonstrating its efficiency and effectiveness across video-text tasks.

Model	Dataset	low only	low+high	MaCP(ours)	4 partitions
RoBERTa-B	MRPC	90.1	89.4	**89.7**	88.9
	CoLA	63.6	64.1	**64.6**	62.9
	QNLI	**92.4**	92.2	**92.4**	92.1
ViT-B	CIFAR100	91.6	**91.7**	**91.7**	91.1
	DTD	74.9	**75.9**	**75.9**	75.2
	EuroSAT	98.9	98.9	**99.1**	98.7
	OxfordPets	93.4	93.3	**93.4**	93.1

Table 8: Performance comparison across datasets with different feature combinations.

4.8 Ablation Study

Expressive Capability: A Comparison of MaCP, FourierFT, and LoRA. To intuitively evaluate the expressive power of our method, we design a simple classification task with a synthetic dataset to simulate a scenario where LoRA encounters performance bottlenecks. Specifically, we specify a 2D center point for each class of data in the 8 classes and randomly add Gaussian noise based on those points to obtain the 2D coordinates of the input. The dataset visualization is shown on the left of Figure 3. We train a single hidden layer with 64×64 units using LoRA ($r = 1$), FourierFT ($n = 128$), and our proposed MaCP ($n = 90$) to fit the synthesized data. Notably, all methods require the same number of trainable parameters, making it a fair comparison.

The experimental results, depicted on the right of Figure 3, highlight significant performance differences among these methods. LoRA exhibits a nonlinear increase in accuracy during the initial training phase but fails to reach 100% accuracy even after 2000 epochs. Instead, it stabilizes with considerable oscillations around 75% accuracy, suggesting that under certain parameter constraints, LoRA encounters performance bottlenecks that hinder its ability to fully capture the underlying data distribution. In contrast, MaCP demonstrates a much smoother learning curve, reaching nearly 100% accuracy within around 450 epochs and maintaining this performance with minimal oscillations.

Ablation Study on Hierarchical Partitioning Schemes. We conducted an ablation study with different partition schemes—"low only," "low+high," and "4 partitions." Table 8 shows that low frequencies are essential for foundational structure, but relying solely on them is insufficient, as high frequencies add crucial details, especially in texture-rich tasks like DTD. While "low+high" performs close to MaCP, ours achieves higher accuracy in tasks like EuroSAT and CoLA. The "4 partitions" scheme performs worst, suggesting that excessive partitioning dilutes key information.

5 Conclusion

We proposed MaCP, a hierarchical cosine projection adaptation method for fine-tuning large language models. By leveraging the energy compaction properties of the Discrete Cosine Transform (DCT), MaCP significantly reduces the number of trainable parameters while maintaining or even enhancing model performance. Extensive experiments across multiple benchmarks demonstrate that MaCP not only outperforms existing state-of-the-art methods in accuracy but also achieves superior computational efficiency and lower storage requirements. These results underscore the practical value of MaCP as a scalable and resource-efficient solution for fine-tuning large-scale models.

6 Limitations

This study has a few limitations. Firstly, the largest model we tested was LLaMA2-13B. Due to time constraints associated with the paper's deadline, we could not extend our experiments to larger models, which could provide further insights into the scalability and effectiveness of MaCP. Secondly, while our method demonstrates strong performance across diverse benchmarks, we did not conduct a detailed analysis of domain-specific tasks, such as biomedical or legal text, which might require further tuning of the frequency partitioning strategy. Lastly, the computational cost of calculating the DCT, while modest, could be optimized further for extremely large-scale models.

References

Seyedarmin Azizi, Souvik Kundu, and Massoud Pedram. 2024. Lamda: Large model fine-tuning via spectrally decomposed low-dimensional adaptation. *arXiv preprint arXiv:2406.12832*.

Zhiguang Cheng, Gaohang Yu, Xiaohao Cai, and Liqun Qi. 2024. A new two-sided sketching algorithm for large-scale tensor decomposition based on discrete cosine transformation. *arXiv preprint arXiv:2404.16580*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. 2014. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. *Preprint*, arXiv:2110.02861.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Max Ehrlich and Larry S Davis. 2019. Deep residual learning in the jpeg transform domain. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3484–3493.

Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. 2024. Parameter-efficient fine-tuning with discrete fourier transform. *arXiv preprint arXiv:2405.03003*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. 2019. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

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. 2022. Lora: Low-rank adaptation of large language models. *International Conference on Learning Representations*.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki Markus Asano. 2023. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. *Technical Report TR-2009*.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara L Berg. 2018. Tvqa: Localized, compositional video question answering. *arXiv preprint arXiv:1809.01696*.

Jie Lei, Licheng Yu, Tamara L Berg, and Mohit Bansal. 2020. Tvr: A large-scale dataset for video-subtitle moment retrieval. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, pages 447–463. Springer.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *EMNLP*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Preprint*, arXiv:1910.13461.

Linjie Li, Yen-Chun Chen, Yu Cheng, Zhe Gan, Licheng Yu, and Jingjing Liu. 2020. Hero: Hierarchical encoder for video+ language omni-representation pre-training. *arXiv preprint arXiv:2005.00200*.

Baohao Liao, Yan Meng, and Christof Monz. 2023a. Parameter-efficient fine-tuning without introducing new latency. *arXiv preprint arXiv:2305.16742*.

Baohao Liao, Shaomu Tan, and Christof Monz. 2023b. Make pre-trained model reversible: From parameter to memory efficient fine-tuning. *Advances in Neural Information Processing Systems*, 36.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. *arXiv preprint arXiv:2004.03829*.

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. *International Conference on Machine Learning*.

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*.

Ryo Nagai, Ryosuke Akashi, and Osamu Sugino. 2020. Completing density functional theory by machine learning hidden messages from molecules. *npj Computational Materials*, 6(1):43.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. 2012. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*.

William H Press. 2007. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918*.

Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. 2021. Do long-range language models actually use long-range context? *EMNLP2021*.

Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. 2022. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5227–5237.

Jianheng Tang, Jiajin Li, Ziqi Gao, and Jia Li. 2022. Rethinking graph neural networks for anomaly detection. In *International Conference on Machine Learning*, pages 21076–21089. PMLR.

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.

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*.

Raja Vavekanand and Kira Sam. 2024. Llama 3.1: An in-depth analysis of the next-generation large language model. -.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. *Preprint*, arXiv:1804.07461.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2023. Transformers in time series: A survey. *IJCAI 2023*.

Kai Xu, Minghai Qin, Fei Sun, Yuhao Wang, Yen-Kuang Chen, and Fengbo Ren. 2020. Learning in the frequency domain. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1740–1749.

Zai Yang and Lihua Xie. 2016. Exact joint sparse frequency recovery via optimization methods. *IEEE Transactions on Signal Processing*, 64(19):5145–5157.

Q Zhang, M Chen, A Bukharin, P He, Y Cheng, W Chen, and T Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. preprint (2023). *arXiv preprint arXiv:2303.10512*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Luowei Zhou, Chenliang Xu, and Jason Corso. 2018. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Xin Zhu, Daoguang Yang, Hongyi Pan, Hamid Reza Karimi, Didem Ozevin, and Ahmet Enis Cetin. 2024. A novel asymmetrical autoencoder with a sparsifying discrete cosine stockwell transform layer for gearbox sensor data compression. *Engineering Applications of Artificial Intelligence*, 127:107322.

Appendix

In this supplementary material, we first provide detailed information about the datasets used in our experiments. Next, we outline the implementation specifics and hyper-parameter settings. We then present additional experimental results that further validate the effectiveness of the proposed MaCP method. Finally, we include examples of instruction tuning to highlight the practical application of our approach MaCP.

7 Details of Datasets

GLUE Benchmark. The General Language Understanding Evaluation (GLUE) benchmark(Wang et al., 2019) is a comprehensive platform designed to evaluate and accelerate progress in natural language understanding (NLU). The benchmark comprises nine tasks that span a variety of NLU challenges such as sentiment analysis, paraphrase detection, linguistic acceptability, natural language inference, and textual similarity. Among these tasks are the Stanford Sentiment Treebank (SST-2), which focuses on binary sentiment classification for movie reviews, and the Microsoft Research Paraphrase Corpus (MRPC), which involves detecting whether two sentences are semantically equivalent. The Corpus of Linguistic Acceptability (CoLA) evaluates a model's ability to distinguish grammatically correct sentences from incorrect ones, reflecting linguistic competence in terms of syntactic judgments. The benchmark's diversity and inclusion of tasks with limited training data encourage the development of models that generalize well across multiple language tasks and genres.

In addition to single-sentence classification tasks, GLUE includes several sentence-pair tasks. The Question Natural Language Inference (QNLI) task is derived from the Stanford Question Answering Dataset (SQuAD). It requires models to determine if a given context sentence contains the answer to a corresponding question. The Recognizing Textual Entailment (RTE) task combines several textual entailment datasets from various domains like news and Wikipedia, testing whether a hypothesis can be logically inferred from a premise. The Semantic Textual Similarity Benchmark (STS-B) measures the similarity between sentence pairs using a regression-based approach, where models predict similarity scores on a continuous scale.

E2E Benchmark. The E2E dataset (Novikova et al., 2017) is designed for training and evaluating end-to-end data-driven natural language generation (NLG) systems within the restaurant domain. It consists of over 50,000 instances and is known for its linguistic complexity, including greater lexical diversity, syntactic variation, and discourse phenomena compared to previous datasets. The evaluation is primarily conducted using five metrics: BLEU, NIST, METEOR, ROUGE-L, and CIDEr. BLEU measures the overlap of n-grams between the generated text and human references, emphasizing precision. METEOR considers synonymy and stemming, providing a more nuanced assessment of text similarity. ROUGE-L focuses on the longest common subsequence to evaluate fluency and structure. CIDEr captures consensus by weighting n-grams based on their relevance in human references, offering a comprehensive measure of output quality.

Instruction Tuning Related Benchmarks The Alpaca dataset (Taori et al., 2023) is a collection of 51K instruction-following examples generated using OpenAI's text-davinci-003. It was created to fine-tune Meta's LLaMA 7B model into a lightweight, instruction-following model named Alpaca. The dataset covers a wide range of tasks, including question-answering, summarization, and classification, enabling the fine-tuned model to exhibit behavior similar to much larger models at a fraction of the cost. A specific example is as follows:

```
 {
    "instructions": Convert the following
temperature from Celsius to Fahrenheit.
    "input": 25 °C
    "output": 25 °C is equal to 77°F.
}
```

MT-Bench (Zheng et al., 2024) is a recently introduced benchmark designed to evaluate the instruction-following capabilities of language foundation models. It consists of a series of open-ended questions aimed at assessing model performance across diverse aspects such as writing, roleplay, reasoning, mathematics, coding, information extraction, STEM, and the humanities. MT-Bench effectively distinguishes these abilities through tailored questions, providing a more comprehensive evaluation. A specific example from the benchmark is provided below.

```
 {
    "Q1": What is the square root of 144?
    "Q2(follow-up)": If you multiply the
answer by 2, what is the result?
    "Solution": Q1: The square root is 12.
Q2: The result is 24.
}
```

Dataset	#Train	#Val	#Test	#Class	Rescaled res.
CIFAR100	45,000	5,000	10,000	100	224×224
DTD	4,060	452	1,128	47	
EuroSAT	16,200	5,400	5,400	10	
OxfordPets	3,312	368	3,669	37	

Table 9: Details about the vision datasets.

Vicuna Eval (Chiang et al., 2023) is a benchmark designed to assess the alignment of large language models (LLMs) with human preferences and serves as the predecessor to MT-Bench. Vicuna Eval evaluates models across a diverse set of topics, including coding, writing, mathematics, counterfactual reasoning, Fermi estimation, common sense, roleplay, knowledge, and generic tasks. It provides a comprehensive framework for understanding how well models align with human expectations in varied scenarios. A specific example from this evaluation is presented below.

```
{
    "question": Explain the concept of
recursion with a simple example.
    "category": programming.
}
```

Image Classification Datasets Tab. 9 provides comprehensive details about four widely-used vision datasets: CIFAR100, DTD, EuroSAT, and OxfordPets. It outlines key information such as the number of training (#Train), validation (#Val), and test (#Test) samples, along with the number of classes (#Class) in each dataset. These datasets span diverse domains, from object recognition (CIFAR100 (Krizhevsky et al., 2009)) to texture classification (DTD (Cimpoi et al., 2014)), satellite image classification (EuroSAT (Helber et al., 2019)), and pet identification (OxfordPets (Parkhi et al., 2012)). The variety in datasets ensures that the models are evaluated across different visual tasks, offering a robust assessment of their performance.

For consistency in model evaluation, all datasets are rescaled to a resolution of 224×224. This standardized input size simplifies comparisons by ensuring that all models receive uniformly sized images, crucial for fair benchmarking. The datasets vary in size and complexity, with CIFAR100 having the largest number of samples (60,000) across 100 classes, while OxfordPets focuses on a more specialized task with 37 classes. The information in this table highlights the diversity of the datasets, each contributing unique challenges to model evaluation.

8 Hyperparamaters

Hyperparameters on GLUE benchmarks Tab. 10 describes the key hyperparameters used in the experiments across different GLUE tasks and model sizes (Base and Large). The table outlines the learning rate schedules, optimizer settings, warmup ratios, and seed values for reproducibility. For both Base and Large models, the AdamW optimizer is used with a linear learning rate schedule and a warmup ratio of 0.06. The frequency bias is set to false, and the frequency coefficient n is fixed at 700 for MaCP unless specified otherwise. Each experiment is run with 5 different seeds {0, 11111, 22222, 33333, 44444}.

For the Base models, the number of training epochs ranges from 30 to 100 depending on the task, with SST-2 requiring the longest training time. The FourierFT and MaCP methods use a higher learning rate for the base models compared to the learning rate used for fine-tuning the head layers.

In contrast, the Large models typically require fewer epochs but use slightly lower learning rates. The batch size remains consistent across both model sizes, set at 32 for all tasks. Additionally, max sequence lengths are adapted to fit the needs of each task, with longer sequences allocated for more complex tasks like CoLA and QNLI.

Hyperparameter settings on E2E benchmark Tab. 11 outlines the hyperparameter configurations used for the medium and large models on the E2E benchmark. Both models are optimized using AdamW with a linear learning rate schedule. The learning rates for MaCP and FourierFT are set to $2E-2$ for the medium model and $5E-2$ for the large model, while the head layers have lower learning rates of $2E-4$ and $1E-4$, respectively. Both models use a batch size of 128, with weight decay values of 0.01 and 0.03 for the medium and large models, respectively. The number of selected frequencies, n, is set to 700 for MaCP and 1000 for FourierFT, while the scaling value α is fixed at 300 for both models. Additionally, label smoothing is applied with a value of 0.1, and training is conducted for 5 epochs.

Hyperparameter Setup for Instruction-Tuning Table 12 summarizes the key hyperparameters used for fine-tuning the LoRA, FourierFT, and MaCP models. Across all methods, the optimizer is AdamW, with a warmup ratio of 0.06, a batch size of 4, and gradient accumulation steps set to 4, providing stability during training. The default training duration is 1 epoch, with certain experiments—such as the motivation example in the intro-

Model	Hyperparameter	STS-B	RTE	MRPC	CoLA	SST-2	QNLI
Both	Optimizer	AdamW					
	LR Schedule	Linear					
	Warmup Ratio	0.06					
	Frequency Bias	False					
	n_{MaCP}	700					
	$n_{FourierFT}$	1000					
	Seeds	{0, 11111, 22222, 33333, 44444}					
Base	Epochs	60	90	30	100	40	40
	Learning Rate (MaCP)	9E-2	9E-2	5E-2	1.2E-1	5E-2	1E-2
	Learning Rate (FourierFT)	9E-2	9E-2	5E-2	1.2E-1	5E-2	1E-2
	Learning Rate (VeRA)	9E-2	9E-2	5E-2	1.2E-1	5E-2	1E-2
	Learning Rate (Head)	9E-3	1.1E-2	6E-3	8E-3	6E-3	1E-3
	Max Seq. Len	512	512	512	512	512	512
	Scaling Value	84	110	141	49	140	29
	Batch Size	32	32	32	32	32	32
Large	Epochs	30	60	30	80	10	30
	Learning Rate (MaCP)	7E-2	8E-2	6E-2	4.3E-2	4.3E-2	6E-2
	Learning Rate (FourierFT)	7E-2	8E-2	6E-2	4.3E-2	4.3E-2	6E-2
	Learning Rate (VeRA)	7E-2	8E-2	6E-2	4.3E-2	4.3E-2	6E-2
	Learning Rate (Head)	1E-3	5E-3	1E-3	1.1E-2	1E-3	5E-3
	Max Seq. Len	512	512	512	256	128	512
	Scaling Value	121	90	120	90	69	69
	Batch Size	32	32	32	128	32	32

Table 10: Hyperparameters used for MaCP across various GLUE tasks.

Hyperparameter	Medium	Large
Optimizer	AdamW	
Learning Rate (MaCP)	2E-2	5E-2
Learning Rate (FourierFT)	2E-2	5E-2
Learning Rate (VeRA)	2E-2	5E-2
Learning Rate (Head)	2E-4	1E-4
Batch Size	128	
Weight Decay	0.01	0.03
n_{MaCP}	700	
$n_{FourierFT}$	1000	
Scaling value α	300	
Epochs	5	
Label Smooth	0.1	
LR Schedule	Linear	

Table 11: Hyperparameter settings on E2E benchmark

Hyperparameter	LoRA	FourierFT	MaCP	VeRA
Optimizer	AdamW			
Warmup Ratio	0.06			
Batch Size	4			
Acc. Steps	4			
Epochs	1 or 2			
n	–	1000	700	–
Scaling Value α	300.0	16.0	16.0	300.0
LR Schedule	Linear			
Learning Rate	3E-2	3E-3	3E-3	3E-3

Table 12: Hyperparameter settings for instruction-tuning configurations.

Hyperparameter	CIFAR100	DTD	EuroSAT	OxfordPets
Epochs	10			
Optimizer	AdamW			
LR Schedule	Linear			
n_{MaCP}	2400			
$n_{FourierFT}$	3000			
α	300.0			
Learning Rate (MaCP)	2E-1	3E-1	2E-1	3E-1
Learning Rate (FourierFT)	2E-1	3E-1	2E-1	3E-1
Learning Rate (VeRA)	2E-1	3E-1	2E-1	3E-1
Learning Rate (Head)	7E-4	1E-3	8E-4	1E-3
Weight Decay	1E-4	7E-5	3E-4	8E-4

Table 13: Hyperparameter setup for image classification.

duction and the ablation study in the supplementary material—using 2 epochs where indicated.

For MaCP, the parameter n is set to 700. The scaling value α differs across methods: it is 300.0 for LoRA, 16.0 for FourierFT, and also 16.0 for MaCP. The learning rate is tuned individually, with LoRA using 3E-2 and both FourierFT and MaCP using a lower rate of 3E-3. All methods follow a linear learning rate schedule.

Hyperparameter setup for image classification
Tab. 13 presents the hyperparameter configurations used for fine-tuning on the CIFAR100, DTD, EuroSAT, and OxfordPets datasets for image classification tasks. The table lists the common settings across these datasets, including the use of the AdamW optimizer, a linear learning rate schedule, and a consistent training setup of 10 epochs. The number of frequency components (n) is set to 2400 for MaCP and 3000 for FourierFT across all datasets.

Hyperparameter	TVQA	How2QA	TVC	YC2C
Epochs	7			
Optimizer	AdamW			
LR Schedule	Q,K			
n_{MaCP}	4500			
$n_{FourierFT}$	6000			
r_{LoRA}	128			
r_{DoRA}	128			
α	300.0			
Learning Rate (MaCP)	3E-4	3E-4	3E-4	3E-4
Learning Rate (FourierFT)	3E-4	3E-4	3E-4	3E-41
Learning Rate (VeRA)	3E-4	3E-4	3E-4	3E-4
Weight Decay	2E-3	2E-5	2E-3	2E-3

Table 14: Hyperparameter setup for video-text understanding

For MaCP and FourierFT, the learning rate varies slightly across datasets, ranging from 2E-1 to 3E-1, while the learning rate for the classification head ranges between 7E-4 and 1E-3. The weight decay is tuned for each dataset, with values ranging from 7E-5 to 1E-4 for DTD and CIFAR100, and slightly higher at 3E-4 and 8E-4 for EuroSAT and Oxford-Pets, respectively.

Hyperparameter setup for video-text understanding Table 14 summarizes the hyperparameter configurations used for fine-tuning the models across four video-text understanding tasks: TVQA, How2QA, TVC, and YC2C. The training process was consistent across all tasks, with each model trained for 7 epochs using the AdamW optimizer. A common learning rate schedule (`Q,K`) was employed for all models.

For the frequency-based methods, the number of selected frequency components, n, was set to 4500 for MaCP and 6000 for FourierFT. For the LoRA and DoRA methods, the rank r was set to 128 for both. The scaling factor α was fixed at 300.0 across all methods.

The learning rate for MaCP, FourierFT, and VeRA was kept at 3E-4 for all tasks, with slight adjustments in the weight decay for different datasets: 2E-3 for TVQA, TVC, and YC2C, and 2E-5 for How2QA.

8.1 Additional Ablation Study

This ablation study further examines the performance of MaCP when trained for two epochs, with results shown in Table 15. Even with extended training, our method consistently surpasses other approaches in both efficiency and performance. For instance, in the LLaMA2-7B model, MaCP achieves an MT-Bench score of 5.27 and a Vicuna score of 7.51 using only 0.045M parameters. In comparison, FourierFT, which uses more parameters (0.064M), falls short in performance. Similarly, on the LLaMA2-13B model, MaCP outperforms all other methods, recording the highest scores of 5.97 on MT-Bench and 8.01 on Vicuna with only 0.056M trainable parameters, compared to FourierFT's 0.08M. These results, highlighted in Table 15, clearly indicate that MaCP remains robust and effective, solidifying its advantage in limited computational resources.

Model	Method	# Tr. Para.	MT-Bench	Vicuna
LLaMA2-7B	LoRA	159.9M	5.24	7.38
	DoRA	163.7M	5.25	7.41
	VeRA	1.6M	5.23	7.47
	FourierFT	0.064M	5.19	7.50
	MaCP	**0.045M**	**5.27**	**7.51**
LLaMA2-13B	LoRA	250.3M	5.85	7.89
	DoRA	264.5M	5.85	7.91
	VeRA	2.4M	5.96	7.86
	FourierFT	0.08M	5.89	7.92
	MaCP	**0.056M**	**5.97**	**8.01**
LLaMA3.1-8B	LoRA	183.3M	5.69	7.52
	DoRA	186.9M	5.72	7.59
	VeRA	1.9M	5.67	7.49
	FourierFT	0.073M	5.70	7.67
	MaCP	**0.05M**	**5.73**	**7.72**

Table 15: Performance comparison of various fine-tuning methods on LLaMA models (LLaMA2-7B, LLaMA2-13B, and LLaMA3.1-8B) evaluated using MT-Bench and Vicuna scores. All models are fine-tuned for 2 epochs and assessed by GPT-4. The scores represent the average across multiple trials, where higher scores indicate better performance. Notably, MaCP consistently outperforms other methods with significantly fewer trainable parameters.

9 Examples of Instruction Tuning

This section highlights practical examples generated by the LLaMA3.1-8B model fine-tuned with FourierFT and MaCP, illustrating the comparative effectiveness of each method. FourierFT uses 146K trainable parameters, while MaCP requires only 100K. Each case includes a prompt, the answers generated by both models, and a review of the responses assessed by GPT-4.

This section presents three case studies comparing models fine-tuned with FourierFT and MaCP. In a restaurant review, MaCP provided more detailed and engaging insights, while FourierFT was concise but less descriptive (Tab. 16). In coding instructions for finding common elements in arrays, both models were correct, but MaCP's explanation was more comprehensive and user-friendly (Tab. 17). Lastly, in film analysis, MaCP offered a deeper, more nuanced critique of plot, character development, and directorial choices, outperforming FourierFT's straightforward evaluation (Tab. 18).

Method	FourierFT	MaCP
Prompt	Write a restaurant review, focusing on the food quality, service, and overall ambiance.	
Answer	The restaurant offered a delightful dining experience with an excellent variety of dishes. The flavors were well-balanced, and the ingredients fresh, though some of the main courses lacked the expected depth. The service was prompt, but it felt somewhat impersonal at times. The ambiance was warm and inviting, with tasteful decor that complemented the overall experience. Overall, it was a pleasant visit with a few areas for improvement.	The restaurant experience was nothing short of exceptional. From the moment we walked in, the attentive staff made sure we felt welcomed. The food was outstanding, with each dish crafted with precision and bursting with flavor. The standout was the chef's special, which delivered a perfect combination of taste and presentation. The ambiance was charming, offering a cozy yet elegant atmosphere. Every detail was carefully thought out, making it a memorable evening overall.
Review	Assistant 1 captured the essence of the dining experience but could have elaborated more on specific dishes or details about the restaurant's atmosphere. The review was concise and clear but left out some critical aspects that would make it more informative and engaging.	Assistant 2 provided a thorough review, highlighting not just the food quality but also the service and ambiance. The attention to detail, such as mentioning the standout dish and how the atmosphere enhanced the overall experience, added depth and made the review more compelling and enjoyable to read.
Score	7.65	**7.80**

Table 16: Comparison of generated responses from models fine-tuned with FourierFT and MaCP on a restaurant review prompt.

Method	FourierFT	MaCP
Prompt	Write a Python program to find the common elements in two arrays without using extra data structures. Provide a step-by-step explanation.	
Answer	**Step-by-Step Explanation:** The process starts by sorting both arrays to simplify the comparison. Sorting helps identify common elements more efficiently. Next, two pointers are initialized, one for each array, starting at the beginning. The algorithm then traverses both arrays in parallel. When the elements at both pointers are equal, the common element is added to the result list. If the first array's element is smaller, the pointer in that array is moved forward. Otherwise, the pointer in the second array is advanced. This continues until one of the pointers exceeds the array's length, at which point all common elements have been found. **Pseudo-code:** Sort(arr1), Sort(arr2). Initialize i, j = 0. While i < len(arr1) and j < len(arr2): If arr1[i] == arr2[j]: Add to result. Else if arr1[i] < arr2[j]: i++. Else: j++. Return result.	**Step-by-Step Explanation:** Begin by sorting both arrays, allowing for easier comparison during traversal. Then, use two pointers to scan both arrays from the start. As the traversal progresses, if the current elements match, they are added to the output list. If the element in the first array is smaller, its pointer is moved forward. If the element in the second array is smaller, its pointer advances. The algorithm concludes when either pointer reaches the end, signaling that all possible common elements have been identified. **Pseudo-code:** Sort(arr1), Sort(arr2). Initialize i, j = 0. Loop until either pointer is out of range: If arr1[i] == arr2[j]: Add to result. If arr1[i] < arr2[j]: Move i forward. Else: Move j forward. Output the result.
Review	This explanation is clear and intuitive, making the approach easy to follow. The importance of sorting and using pointers is well-highlighted. However, while the explanation is technically sound, it could benefit from addressing potential edge cases like handling empty arrays or duplicates.	This explanation is detailed and user-friendly, with a strong emphasis on clear decision-making within the algorithm. The pointers are well-explained, and the pseudo-code is both concise and logical. The approach effectively handles typical scenarios, but more depth on edge cases could further improve it.
Score	7.79	**7.97**

Table 17: Comparison of generated responses from models fine-tuned with FourierFT and MaCP on a coding instruction prompt.

Method	FourierFT	MaCP
Prompt	Provide a detailed analysis of the film's plot structure, character development, directorial choices, and the role of cinematography in reinforcing the central themes.	
Answer	The film expertly weaves together a multi-layered plot that explores themes of redemption, identity, and the consequences of past actions. The narrative unfolds in a non-linear fashion, employing flashbacks and parallel timelines that gradually reveal the backstory of the characters. This intricate structure keeps the audience engaged, constantly piecing together clues while deepening the emotional stakes. The character development is a highlight, particularly the protagonist, whose transformation from a morally ambiguous figure to someone grappling with guilt and seeking redemption is portrayed with remarkable depth. The supporting characters, though less developed, provide crucial counterpoints that enhance the protagonist's journey. The director's use of long takes and carefully framed shots adds a sense of intimacy and tension, making the audience feel like silent observers within the story. The cinematography further complements this by using stark contrasts between light and shadow to symbolize the internal conflicts of the characters. The film's visual language is rich with symbolism, from recurring motifs like broken mirrors to the deliberate use of desaturated color palettes during scenes of despair. However, the third act, while visually stunning, slightly overextends itself, relying too heavily on exposition. Despite this, the film's ability to blend a complex narrative with powerful imagery makes it a standout example of modern cinema.	The film's narrative is a compelling exploration of complex human emotions, weaving a story that is both introspective and expansive in scope. The plot structure is intricately layered, using a blend of real-time events and flashbacks to gradually reveal the true motivations behind each character's actions. The protagonist's arc is especially well-crafted, showing a slow but steady transformation driven by a combination of external conflicts and internal struggles. This gradual change is mirrored in the pacing, which starts off deliberately slow, allowing the audience to fully grasp the gravity of each decision made by the characters. The secondary characters, while often playing subtle roles, are given moments that add depth to the overall narrative. Directorial choices play a significant role in maintaining the film's atmospheric tone. Through the use of visual storytelling, the director skillfully conveys the tension and unresolved emotions between characters without resorting to overt dialogue. The cinematography is not just a backdrop but an active participant in the storytelling. For example, the frequent use of close-up shots during moments of silence emphasizes the unspoken emotions that linger beneath the surface. The decision to film key scenes during golden hour creates a visual contrast between the warmth of the setting sun and the cold, stark reality of the characters' situations. The careful choreography of camera movements during action sequences adds a layer of realism that pulls the audience further into the narrative. While the film's conclusion leaves some plot threads unresolved, this ambiguity feels intentional, prompting the viewer to reflect on the underlying themes long after the credits roll.
Review	The analysis provided a thorough overview of the film's plot structure and character development. While it accurately identified key themes and offered insight into directorial choices, it occasionally lacked specific examples that would make the points more concrete. The discussion on cinematography was solid, but a more detailed breakdown of particular scenes or camera techniques could have strengthened the overall argument. Overall, this review was clear, relevant, and informative, but some areas could benefit from more in-depth analysis and illustrative examples.	This analysis delivered a comprehensive breakdown of the film's narrative layers, seamlessly integrating observations on character development, pacing, and the director's stylistic choices. The reviewer highlighted the interaction between visual storytelling and thematic content, using specific examples to support each point. The detailed exploration of the protagonist's transformation was particularly effective, demonstrating a nuanced understanding of the film's emotional core. The cinematography discussion not only described the techniques used but contextualized their impact within the broader narrative framework. The thoroughness and attention to detail in this review made it both insightful and engaging.
Score	7.91	**8.09**

Table 18: Comparison of generated responses from models fine-tuned with FourierFT and MaCP on an extended movie analysis prompt.