Towards More Efficient Post-training via Fourier Domain Adapter Framework

Yijia Fan, Jusheng Zhang, Keze Wang[†]
Sun Yat-sen University

[†]Corresponding author: kezewang@gmail.com

Abstract

We introduce Fourier Domain Adapter (FDA), a novel and parameter-efficient framework for fine-tuning large-scale pre-trained language models. FDA reparameterizes the core projection operation of the adapter module directly in the Fourier domain. This involves transforming the input features via discrete Fourier transform (DFT), applying sparse learnable complex modulations in frequency space, and then back-transforming via inverse DFT, supplemented by highly compact auxiliary linear layers. This approach significantly reduces the number of trainable parameters while enhancing the model's ability to capture salient frequency-based semantic information. Comprehensive experiments on GLUE, E2E NLG, and instruction tuning benchmarks show that our FDA consistently outperforms existing parameter-efficient fine-tuning (PEFT) methods. It can achieve better performance with nearly 100x fewer training parameters than traditional fine-tuning methods such as LoRA and AdapterH. Our results demonstrate that FDA is a robust and efficient solution for developing efficient and powerful language models.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2024; Zhang et al., 2025d,b) have revolutionized numerous areas of natural language processing by demonstrating an exceptional ability to store vast amounts of knowledge during pre-training and effectively recall this knowledge during inference. However, despite these capabilities, LLMs frequently "generating inaccurate or outdated information (Huang et al., 2025; Farquhar et al., 2024; Zhang et al., 2025c)," generating inaccurate or outdated information. To address this, the academic community has introduced fine-tuning methods. Yet, full fine-tuning to adapt LLMs to specific downstream tasks or update their internal knowledge is often prohibitively expensive due to its substantial computational and time costs. Consequently, Parameter-Efficient Fine-Tuning (PEFT) methods (Han et al., 2024; Ding et al., 2023) have emerged, aiming to update target knowledge or behaviors with minimal additional overhead while preserving the model's original capabilities.

Broadly, current PEFT methods primarily focus on achieving efficient adaptation by modifying a small number of parameters or introducing small additional modules while keeping the majority of LLM parameters frozen. In this context, **Adapter** (Houlsby et al., 2019; Tang et al., 2025) modules and **Low-Rank Adaptation (LoRA)** (Hu et al., 2021) have become two highly influential mainstream paradigms. Adapters typically insert small feed-forward network (FFN) modules between the layers of a pre-trained model. These modules comprise a down-projection layer with an activation function and an up-projection layer without one. LoRA, on the other hand, indirectly modifies model behavior with a small number of trainable parameters by decomposing the updates to pre-trained weights into the product of two low-rank matrices. Both strategies aim to significantly reduce the number of trainable parameters during the fine-tuning process. Despite the remarkable success of these methods in parameter efficiency, existing mainstream PEFT paradigms still face a **critical bottleneck** in balancing extreme parameter compression with the maintenance of high performance. Specifically, although traditional adapters are termed "lightweight," their projection layers rely on **dense weight matrices**. This fundamental design means that even with small adapter dimensions, their total parameter count (often reaching millions) poses a severe deployment challenge for edge devices, which are highly sensitive to memory and computational resources. Conversely, LoRA achieves substantial parameter reduction through **low-rank approximation**, but this inherent low-

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 6175–6193
November 4-9, 2025

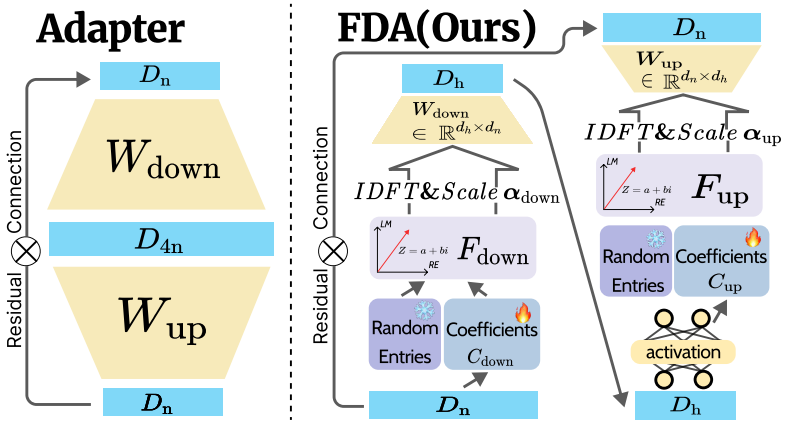

Figure 1: Comparison of a traditional adapter (left) and our Fourier Domain Adapter (FDA, right). (Left) The traditional adapter employs down-projection (W_{down}), non-linearity, and up-projection (W_{up}) layers, with a residual connection. (Right) FDA reparameterizes projections in the Fourier domain. An input (D_n) is transformed into the frequency domain, modulated by a sparsely parameterized learnable filter F_{down} (driven by coefficients C_{down} for selected frequency entries), scaled by α_{down} after an Inverse DFT (IDFT), and then passed through a compact linear layer W_{down} to an intermediate dimension D_h. Following a non-linear activation, a symmetric up-projection path (employing F_{up} from C_{up}, α_{up}, and W_{up}) maps D_h back to D_n. Both adapter types utilize residual connections and are typically inserted between existing layers of a pre-trained model.

rank constraint itself may limit its capacity to fully express complex state transitions. When tackling tasks that require capturing high-frequency signals or subtle semantic differences, the low-rank assumption can become a limiting factor for model performance, leading to suboptimal outcomes in specific scenarios or even sacrificing some of the inherent powerful representation potential of LLMs.

To address the aforementioned drawbacks, i.e., the parameter redundancy of traditional adapters and the potential representational limitations of LoRA, we explore a novel parameterization approach. Instead of being confined to optimizing the scale of dense projections or relying on fixed low-rank decompositions, we attempt to **parameterize the core projection operations of adapters in the Fourier domain** to map input features to the frequency domain via Fourier transforms and learn adaptive transformations within this frequency space, thereby replacing traditional dense weight matrices while enhancing the adapter's sensitivity to frequency components. We posit that frequency-domain representations have the potential to efficiently capture multi-scale, periodic, or high-frequency features with fewer parameters, offering a way to overcome the limitations of traditional spatial-domain parameterization. Notably, in our proposed architecture, rigorous experimental validation has shown that the non-linear activation function in the down-projection path is crucial for maintaining model performance and is retained.

Our **main** contributions are: i) The introduction of the Fourier Domain Adapter (FDA), a novel Parameter-Efficient Fine-Tuning (PEFT) paradigm that reparameterizes adapter projection layers directly in the Fourier domain; ii) Endowing the adapter with significantly enhanced frequency perception and adaptive capabilities. This is achieved by its unique Fourier domain parameterization, which allows for targeted, sparse modulation of frequency components to effectively address the representational limitations of prior PEFT methods; iii) The comprehensive experiments demonstrate that FDA consistently outperforms existing parameter-efficient fine-tuning methods on multiple benchmarks, including GLUE, E2E NLG, and instruction fine-tuning while requiring only a few trainable parameters. For example, when fine-tuning large models such as LLaMA3-70B (such as the E2E NLG task), FDA requires more than 151 times fewer trainable parameters than the traditional AdapterH and 112 times less than LoRA.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT) methods have gained widespread application in the adaptive adjustment of large-scale pre-trained language models in recent years. Traditional full-parameter fine-tuning methods (Liu et al., 2019a; Lv et al., 2024; Han et al., 2016; Zhang et al.,

2025a) require updating a large number of model parameters when dealing with specific tasks, leading to high computational and storage costs. To address this issue, researchers have proposed various PEFT methods, such as Adapters (Houlsby et al., 2019) and LoRA (Hu et al., 2021). Adapters insert lightweight adapter modules between the layers of the model, fine-tuning only these new parameters can significantly reduce the number of parameters required for fine-tuning. LoRA reduces the scale of parameter updates through low-rank matrix decomposition.

Frequency Domain Enhancement and Fourier Transform Frequency domain analysis, successful in computer vision (Mallat, 1989; Xu et al., 2020; Li et al., 2025; Fu et al., 2025b), is gaining traction in NLP (Verma and Pilanci, 2024). By transforming text signals into the frequency domain, these methods better capture high and low-frequency features, improving pattern understanding. Recent work (He et al., 2023; Hua et al., 2025; Fu et al., 2025a) has integrated Fourier transforms into language models, enhancing multi-frequency semantic modeling (Jin et al., 2024) and showing benefits in cross-domain and low-resource scenarios. Recent works (Gries and Divjak, 2012; Tamkin et al., 2020) show that key semantic information concentrates in specific frequency bands, with methods like (Gao et al., 2024) decomposing inputs to better capture multi-frequency components. However, current approaches have not fully leveraged frequency domain structures for semantic representation, making the optimization of these techniques an important research direction.

Fourier-Based Parameterization of Linear Transformations. Dense matrix multiplications in neural networks are a major parameter bottleneck (Yu et al., 2017; Schäfer et al., 2020; Abboud et al., 2020), particularly in large models, driving research into their efficient compression. While pruning (LeCun et al., 1989; Han et al., 2015) and quantization (Jacob et al., 2017) offer solutions, Fourier transforms provide a compelling strategy for efficiently parameterizing these linear operations in the frequency domain. Such approaches approximate weight matrices with fewer Fourier domain parameters, for example, by learning sparse or structured frequency coefficients. Methods(Gao et al., 2024; Borse et al., 2024) like FourierFT show Fourier analysis can significantly reduce matrix multiplication parameters, often by decomposing transformations via operations on fewer frequency-domain components. Our work leverages these Fourier strategies to re-parameterize adapter module projection layers, creating highly parameter-efficient fine-tuning solutions for LLMs.

3 Methodology

3.1 FDA Overall Architecture

Similar to traditional adapter modules (as illustrated in, e.g., the left panel of Figure 1), our proposed FDA module (depicted in, e.g., the right panel of Figure 1) is designed to be inserted between the layers of a pre-trained LLM. FDA receives the input hidden state $X \in \mathbb{R}^{B\times N\times d_{model}}$ from that layer (or a specific sub-module, such as after an attention layer or a feed-forward network layer), where B is the batch size, N is the sequence length, and d_{model} represents the model's original hidden feature dimension. FDA processes the input features through a parameter-efficient down-projection, a non-linear activation function, and an equally parameter-efficient up-projection. Finally, the processed result is added back to the original input X via a residual connection. Notably, the FDA operates independently on the representation of each token in the sequence. Therefore, for clarity, the descriptions in the following subsections will primarily focus on the input representation $x \in \mathbb{R}^{d_{model}}$ for a single token. The core innovation of FDA lies in how its down-projection and up-projection operations are efficiently parameterized in the Fourier domain.

3.2 Fourier-Parameterized Down-Projection

The primary function of the down-projection module is to map the input feature $x \in \mathbb{R}^{d_{model}}$ from the model's original hidden dimension d_{model} to a lower intermediate hidden dimension d_h. This process is meticulously designed through several steps to maximize parameter efficiency and leverage frequency-domain properties. First, the input feature x (specifically, operating along its d_{model} dimension) is transformed into the frequency domain using a 1D Discrete Fourier Transform (DFT):

$$x_f = \text{DFT}(x) \tag{1}$$

where $x_f \in \mathbb{C}^{d_{model}}$ is the complex representation of the input feature x in the frequency domain. Second, a learnable modulation filter F_{down} is applied in the frequency domain. This filter is not learned densely but is sparsely parameterized as follows.

We define R_{down} as a pre-defined set of k_{down} frequency indices, fixed at model initialization by uniformly randomly sampling k_{down} indices without replacement from the interval $[0, d_{model}-1]$. These indices specify which particular frequency components are targeted for learning and adjustment. Correspondingly, $C_{down} \in \mathbb{C}^{k_{down}}$ is a set of trainable complex coefficients, equal in number to the selected frequencies in R_{down}, with each coefficient corresponding to a specific frequency in R_{down}. The complete modulation filter $F_{down} \in \mathbb{C}^{d_{model}}$ is constructed such that for each index j in R_{down} (let its value be idx_j), $F_{down}[idx_j]$ takes its value from the trainable coefficient $C_{down}[j]$; for all other frequency indices not selected by R_{down}, the value of F_{down} at these positions is fixed to 1.0. This design allows the filter to primarily perform dynamic amplitude and phase modulation on the selected k_{down} frequencies, while other frequency components are, by default, passed through. Such a mechanism aims to preserve most of the original signal's information structure while applying targeted, efficient adaptive adjustments. This modulation filter is then applied element-wise to x_f:

$$x'_f = x_f \odot F_{down}(C_{down}, R_{down}) \tag{2}$$

where $\odot$ denotes the Hadamard (element-wise) product. Third, the modulated frequency-domain representation x'_f is transformed back to the spatial domain using an Inverse Discrete Fourier Transform (IDFT). The transformed signal is then globally scaled by a learnable scalar parameter α_{down}:

$$h_{scaled} = \alpha_{down} \cdot \text{IDFT}(x'_f) \tag{3}$$

where $h_{scaled} \in \mathbb{R}^{d_{model}}$. Fourth, this scaled spatial-domain representation h_{scaled} is passed through a standard, yet parameter-wise small, linear projection layer $\mathbf{W}_{down} \in \mathbb{R}^{d_h \times d_{model}}$. This layer maps h_{scaled} from the d_{model} dimension to the target intermediate hidden dimension d_h:

$$H_{down} = \mathbf{W}_{down} h_{scaled} \tag{4}$$

The parameter count of this linear layer $\mathbf{W}_{down}$ (i.e., $d_h \times d_{model}$) is kept efficient by choosing $d_h \ll d_{model}$. It primarily serves for final dimensionality alignment and smoothing of the feature representation, while the core, complex feature transformations and parameterization are achieved with lower parameter cost through the preceding frequency-domain operations.

In summary, the entire down-projection process can be expressed as:

$$H_{down} = \underbrace{\mathbf{W}_{down}}_{\substack{\text{Small Output} \\ \text{Linear Layer}}} \left(\underbrace{\alpha_{down}}_{\text{Learnable Scalar}} \cdot \underbrace{\text{IDFT}\left(\underbrace{\text{DFT}(x)}_{\substack{\text{Input to} \\ \text{Freq. Domain}}} \odot \underbrace{F_{down}(C_{down}, R_{down})}_{\substack{\text{Learnable Sparse Freq.} \\ \text{Modulation (Defaults to Pass-Through)}}} \right)}_{\text{Core Fourier Transform and Modulation Path}} \right) \tag{5}$$

where the output $H_{down} \in \mathbb{R}^{d_h}$ is the down-projected and transformed hidden representation.

3.3 Non-linear Activation

To introduce the necessary non-linear expressive power, enabling FDA to learn more complex functions, we pass the output of the down-projection, H_{down}, through a non-linear activation function $\sigma(\cdot)$. In this study, we employ GELU (Gaussian Error Linear Unit) as the activation function, consistent with choices in many modern Transformer models:

$$H_{act} = \sigma(H_{down}) = \text{GELU}(H_{down}) \tag{6}$$

where $H_{act} \in \mathbb{R}^{d_h}$. We found that retaining this non-linear activation is crucial for maintaining the model's performance on downstream tasks.

3.4 Fourier-Parameterized Up-Projection

The up-projection module maps the activated intermediate hidden representation $H_{act} \in \mathbb{R}^{d_h}$ from the dimension d_h back to the model's original hidden dimension d_{model}. Its overall structure is symmetric to the down-projection module and similarly utilizes the Fourier domain for parameter-efficient transformation.

First, the activated features H_{act} are transformed into the frequency domain (along the d_h dimension):

$$(H_{act})_f = \text{DFT}(H_{act}) \tag{7}$$

where $(H_{act})_f \in \mathbb{C}^{d_h}$.

Second, selective frequency modulation is applied:

$$(H'_{act})_f = (H_{act})_f \odot F_{up}(C_{up}, R_{up}) \tag{8}$$

Here, $F_{up} \in \mathbb{C}^{d_h}$ is the modulation filter for the up-projection path, constructed from trainable complex coefficients $C_{up} \in \mathbb{C}^{k_{up}}$ (corresponding to k_{up} fixed frequency indices in R_{up}, randomly selected from $[0, d_h - 1]$) and default values of 1.0 at other frequency positions.

Third, the modulated signal is transformed back to the spatial domain and scaled:

$$h'_{up} = \alpha_{up} \cdot \text{IDFT}((H'_{act})_f) \tag{9}$$

where α_{up} is the learnable scaling factor for the up-projection path, and $h'_{up} \in \mathbb{R}^{d_h}$.

Fourth, a final linear projection is applied:

$$H_{up} = \mathbf{W}_{up} h'_{up} \tag{10}$$

where $\mathbf{W}_{up} \in \mathbb{R}^{d_{model} \times d_h}$ is another (parameter-wise small) linear projection layer, responsible for restoring the features to the original dimension d_{model}. The entire up-projection process can be summarized as:

$$H_{up} = \underbrace{\mathbf{W}_{up}}_{\substack{\text{Small Output} \\ \text{Linear Layer}}} \left(\underbrace{\alpha_{up}}_{\text{Learnable Scalar}} \cdot \underbrace{\text{IDFT}\left(\underbrace{\text{DFT}(H_{act})}_{\substack{\text{Activated Features} \\ \text{to Freq. Domain}}} \odot \underbrace{F_{up}(C_{up}, R_{up})}_{\substack{\text{Learnable Sparse Freq.} \\ \text{Modulation (Defaults to Pass-Through)}}} \right)}_{\text{Core Fourier Transform and Modulation Path}} \right) \tag{11}$$

where $H_{up} \in \mathbb{R}^{d_{model}}$ is the final output of the FDA module before the residual connection.

4 Experiments

We evaluate FDA fine-tuned NLP models across three perspectives: (1) Natural Language Understanding (NLU) tasks on the GLUE benchmark (Wang et al., 2019) with RoBERTa (Base & Large) (Liu et al., 2019b), (2) Natural Language Generation (NLG) tasks on the E2E NLG dataset (Dušek et al., 2020) using GPT2-Small (Radford et al., 2019), DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025), LLaMA2-7B (Meta-AI, 2023), LLaMA3-8B (Meta-AI, 2024)and LLaMA3-70B(Meta-AI, 2024), and (3) instruction tuning tasks on MT-Bench (Zheng et al., 2023), Vicuna Eval (Chiang et al., 2023), BBH (Suzgun et al., 2022), MATH (Hendrycks et al., 2021), and Alpaca (Taori et al., 2023) with DeepSeek-R1-Distill-Qwen-1.5B, LLaMA2-7B, Qwen2-7B, and LLaMA3-8B.For a detailed introduction to the dataset, see the Supplementary Materials section in the Appendix E.1.In addition, we also designed frequency perception experiments and ablation experiments to test the specific frequency performance of the FDA fine-tuned model and the impact of each component on the FDA model. All experiments were performed on eight A100 GPUs. Throughout all experiments, the intermediate dimension d_h of the FDA modules was configured to $d_{model}/4$, where d_{model} denotes the input hidden dimension of the pre-trained model. For instance, in the case of RoBERTa, where $d_{model} = 768$, the intermediate dimension d_h was set to $768/4 = 192$.

4.1 Compared PEFT Methods

We compare the FDA method with currently popular parameter-efficient fine-tuning (PEFT) methods, using the experimental settings of each respective method. The models involved in the comparison include:**Full Parameter Fine-tuning (FF)**: All parameters are updated, leading to high computational and storage costs. • **AdapterH**: Inserts an adapter layer between self-attention and the feedforward network. • **AdapterL(Lin et al., 2020)**: Adds a lightweight adapter layer only after the MLP module. • **AdapterP**: Optimizes adapter placement after the feedforward layer for better task adaptation. • **Compacter(Mahabadi et al., 2021)**: Uses low-rank parameterization to reduce storage and computation. • **Parallel Adapter(Huh et al., 2024)**: Uses parallel adapters to enhance inference efficiency. • **LoRA**: Fine-tunes low-rank matrices to reduce the parameter updates during training. • **FourierFT**: Replaces low-rank approximations with Fourier transforms to cut down parameters. Please note that due to model adaptation and dataset loading issues, we may choose different comparison models for different tasks.

4.2 Natural Language Understanding

Experimental Setup The baseline models are pre-trained RoBERTa Base (12 layers, 768 hidden units) and RoBERTa Large (24 layers, 1024 hidden units), using their official configurations. During fine-tuning, we adopt our proposed Fourier Domain Adapter (FDA) modules, which are inserted between the Transformer layers and feed-forward layers, with a total of 4 adapter layers. FDA significantly reduces trainable parameters compared to traditional adapters. Additionally, the weights of all structures, except for the classification head, are frozen during fine-tuning. The specific hyperparameter settings for the experiments are provided in Appendix A. We evaluate the fine-tuned models on their comprehension ability across eight tasks: CoLA, SST-2, MRPC, QQP, QNLI, RTE, STS-B, and WNLI. For specific training time comparisons, see the supplementary materials section in the appendix F.

Experimental Results The exceptional efficiency and effectiveness of FDA are strikingly demonstrated in Table 1. Our approach consistently achieves state-of-the-art or highly competitive performance across the GLUE benchmark tasks, while operating with an extraordinarily minimal num-

Method	#Paras	Datasets							
		CoLA (MCC)	SST-2 (Acc.)	MRPC (Acc.)	QQP (Acc.)	QNLI (Acc.)	RTE (Acc.)	STS-B (PCC)	WNLI (PCC)
FF	125M	$64.5_{\pm 0.3}$	$96.1_{\pm 0.2}$	$91.8_{\pm 0.2}$	$95.3_{\pm 0.3}$	$94.3_{\pm 0.3}$	$82.8_{\pm 0.3}$	$93.6_{\pm 0.4}$	$66.3_{\pm 0.3}$
AdapterH	0.6M	$60.8_{\pm 0.4}$	$94.2_{\pm 0.1}$	$88.5_{\pm 1.1}$	$93.5_{\pm 0.3}$	$93.1_{\pm 0.1}$	$71.5_{\pm 1.2}$	$89.7_{\pm 0.3}$	$64.2_{\pm 0.5}$
AdapterL	0.6M	$62.6_{\pm 0.9}$	$94.7_{\pm 0.3}$	$88.4_{\pm 0.1}$	$94.8_{\pm 0.2}$	$93.0_{\pm 0.2}$	$75.9_{\pm 0.2}$	$90.3_{\pm 0.1}$	$64.5_{\pm 0.3}$
AdapterP	0.3M	$63.4_{\pm 1.2}$	$95.1_{\pm 0.2}$	$89.7_{\pm 0.7}$	$93.0_{\pm 0.5}$	$93.3_{\pm 0.3}$	$78.4_{\pm 0.8}$	$91.5_{\pm 0.2}$	$65.0_{\pm 0.4}$
Compacter	0.3M	$62.0_{\pm 0.6}$	$94.5_{\pm 0.2}$	$88.7_{\pm 0.5}$	$92.3_{\pm 0.4}$	$93.1_{\pm 0.2}$	$81.0_{\pm 0.6}$	$90.5_{\pm 0.2}$	$64.8_{\pm 0.2}$
Parallel Adapter	1.2M	$61.1_{\pm 0.3}$	$94.3_{\pm 0.5}$	$89.5_{\pm 0.5}$	$94.7_{\pm 0.4}$	$92.2_{\pm 0.5}$	$78.7_{\pm 0.7}$	$91.1_{\pm 0.6}$	$64.9_{\pm 0.1}$
LoRA	0.3M	$63.8_{\pm 1.6}$	$94.2_{\pm 0.3}$	$90.0_{\pm 0.8}$	$93.5_{\pm 0.6}$	$92.2_{\pm 0.1}$	$79.1_{\pm 0.5}$	$92.8_{\pm 0.4}$	$65.2_{\pm 0.3}$
FourierFT	0.024M	$62.3_{\pm 1.4}$	$94.2_{\pm 0.2}$	$90.3_{\pm 0.3}$	$92.0_{\pm 0.4}$	$91.7_{\pm 0.4}$	$78.4_{\pm 1.6}$	$91.0_{\pm 0.4}$	$66.0_{\pm 0.5}$
FDA (Ours)	**0.011M**	**65.7**$_{\pm 0.3}$	**97.5**$_{\pm 0.2}$	**92.8**$_{\pm 0.3}$	**97.6**$_{\pm 0.2}$	**95.8**$_{\pm 0.2}$	**83.1**$_{\pm 0.4}$	**95.2**$_{\pm 0.3}$	**68.1**$_{\pm 0.3}$
FF	356M	$69.1_{\pm 0.2}$	$96.9_{\pm 0.3}$	$92.3_{\pm 0.5}$	$92.2_{\pm 0.4}$	$95.7_{\pm 0.3}$	$89.2_{\pm 0.5}$	$93.1_{\pm 0.3}$	$67.0_{\pm 0.4}$
AdapterH	1.8M	$68.3_{\pm 1.0}$	$96.1_{\pm 0.3}$	$90.2_{\pm 0.7}$	$91.8_{\pm 0.5}$	$94.8_{\pm 0.2}$	$83.8_{\pm 2.9}$	$92.1_{\pm 0.7}$	$65.5_{\pm 0.3}$
AdapterL	1.8M	$67.8_{\pm 2.5}$	$96.6_{\pm 0.2}$	$89.7_{\pm 1.2}$	$91.5_{\pm 0.4}$	$94.8_{\pm 0.3}$	$80.1_{\pm 2.9}$	$91.9_{\pm 0.4}$	$65.8_{\pm 0.2}$
AdapterP	0.9M	$66.5_{\pm 0.4}$	$96.2_{\pm 0.3}$	$88.7_{\pm 2.9}$	$91.2_{\pm 0.6}$	$94.7_{\pm 0.2}$	$83.4_{\pm 1.1}$	$91.0_{\pm 1.7}$	$65.3_{\pm 0.4}$
Compacter	0.9M	$66.3_{\pm 2.0}$	$96.3_{\pm 0.5}$	$87.7_{\pm 1.7}$	$91.0_{\pm 0.5}$	$94.7_{\pm 0.2}$	$88.4_{\pm 2.9}$	$91.5_{\pm 0.5}$	$65.0_{\pm 0.3}$
Parallel Adapter	4.8M	$68.2_{\pm 1.9}$	$96.2_{\pm 0.5}$	$90.2_{\pm 1.0}$	$91.8_{\pm 0.4}$	$94.8_{\pm 0.3}$	$85.2_{\pm 1.1}$	$92.3_{\pm 0.5}$	$66.0_{\pm 0.2}$
LoRA	0.8M	$67.1_{\pm 1.4}$	$96.0_{\pm 0.2}$	$91.5_{\pm 0.3}$	$91.5_{\pm 0.4}$	$94.4_{\pm 0.4}$	$87.4_{\pm 1.6}$	$91.9_{\pm 0.4}$	$66.2_{\pm 0.3}$
FourierFT	0.048M	$68.5_{\pm 1.2}$	$95.3_{\pm 0.3}$	$91.2_{\pm 0.4}$	$92.0_{\pm 0.5}$	$94.9_{\pm 0.3}$	$87.5_{\pm 1.4}$	$92.5_{\pm 0.5}$	$66.8_{\pm 0.4}$
FDA (Ours)	**0.014M**	**70.2**$_{\pm 0.2}$	**99.3**$_{\pm 0.1}$	**94.2**$_{\pm 0.2}$	**94.5**$_{\pm 0.3}$	**96.9**$_{\pm 0.2}$	**91.2**$_{\pm 0.3}$	**94.9**$_{\pm 0.2}$	**68.8**$_{\pm 0.3}$

Table 1: Performance of various fine-tuning methods with RoBERTa Base (upper part) and RoBERTa Large (lower part) models on 8 datasets of the GLUE benchmark. We report the Matthew's correlation coefficient (MCC) for CoLA, Pearson correlation coefficient (PCC) for STS-B and WNLI, and accuracy (Acc.) for all the remaining tasks. We report the median result of 5 runs, each using different random seeds. The best results for each dataset are shown in bold. Higher is better for all metrics in 8 datasets.

ber of trainable parameters. For RoBERTa-Base, FDA, using only about 0.011M parameters, delivers leading results such as a Matthews Correlation Coefficient of 65.7 on CoLA, 97.6% accuracy on QQP, and a Pearson Correlation Coefficient of 68.1 on WNLI. This parameter count is drastically lower than traditional AdapterH (0.6M) and LoRA (0.3M), and is less than half that of FourierFT (0.024M), yet FDA often surpasses these methods in performance. Moreover, Table 1 reveals that FDA consistently outperforms Full Fine-tuning across these tasks. We are particularly enthused by this finding, as it suggests that FDA's intrinsic frequency-aware architecture might promote better generalization by mitigating overfitting, a common challenge in traditional full fine-tuning. This profound parameter efficiency extends to RoBERTa-Large, where FDA requires a mere approximately 0.014M parameters. These results underscore FDA's innovative architecture, which strategically allocates a $2d_{model}$ budget for its core Fourier transformations across all adapter modules, complemented by highly compact linear layers (e.g., $d_h = 1$) for fine-grained adjustments. This design facilitates potent LLM fine-tuning with minimal parametric overhead, significantly broadening the accessibility for deploying advanced models.

4.3 Natural Language Generation

Experimental Setup. We evaluate the natural language generation capability of FDA fine-tuned models. The models are trained for 30 epochs, and results are recorded from the best test set performance. Specific hyperparameter settings are detailed in Appendix A. For FDA, we consistently apply an architecture with 4 adapter layers, and the internal parameters are set following the principles outlined in Section 3, leading to ultra-low trainable parameter counts.

Experimental Results. The performance of our Fourier Domain Adapter (FDA) on the End-to-End NLG Benchmark is detailed in Table 2. FDA demonstrates a remarkable combination of superior generation quality and unparalleled parameter efficiency across all tested models. For instance, with GPT-2 Small, FDA achieves a BLEU score of 68.81 and a METEOR score of 47.73 using an exceptionally scant 0.011M trainable parameters. This pattern of excellence extends to larger models: FDA on DeepSeek R1-1.5B (0.029M params) and LLaMA2-7B (0.057M params) consistently sets new state-of-the-art results for PEFT methods. This outstanding efficiency and performance extend impressively to very large models, as demonstrated with LLaMA3-70B where FDA achieves top scores

Table 2: Performance comparison of different methods on the end-to-end natural language generation benchmark. FDA denotes our Fourier Domain Adapter. We ran 10 experiments with different random seeds and recorded the best test set performance.

Model	Method	# Trainable Parameters	BLEU	NIST	METEOR	ROUGE-L	CIDEr
GPT-2 Small	FF	123.65M	67.63	8.42	46.71	71.68	2.41
	AdapterH	0.12M	66.11	8.35	44.39	68.75	2.39
	AdapterL	0.12M	66.77	8.21	44.16	70.13	2.28
	FourierFT	0.017M	66.36	8.37	45.85	70.44	2.34
	LoRA	0.13M	66.94	8.32	46.26	70.97	2.33
	FDA(Ours)	**0.011M**	**68.81**	**8.62**	**47.73**	**73.00**	**2.46**
Deepseek R1- 1.5B	FF	1.5B	87.72	9.78	68.93	90.14	3.23
	AdapterH	1.63M	86.34	9.66	68.15	88.23	2.98
	AdapterL	1.63M	86.75	9.67	67.76	89.48	3.13
	FourierFT	0.15M	86.42	9.62	67.97	89.45	2.92
	LoRA	1.21M	87.03	9.66	68.26	88.93	3.15
	FDA(Ours)	**0.029M**	**89.21**	**9.96**	**70.31**	**91.90**	**3.31**
LLaMA2 7B	FF	6.74B	73.16	9.43	51.12	74.91	2.77
	AdapterH	7.27M	72.72	9.26	50.33	73.94	2.62
	AdapterL	7.27M	72.36	9.15	50.17	73.88	2.52
	FourierFT	0.82M	72.52	9.27	49.73	73.78	2.74
	LoRA	5.37M	72.41	9.32	50.27	74.38	2.67
	FDA(Ours)	**0.057M**	**74.76**	**9.55**	**51.94**	**76.31**	**2.82**
LLaMA3 70B	FF	≈70B	86.53	10.12	65.24	88.13	4.36
	AdapterH	17.46M	83.47	9.63	62.76	85.18	4.04
	AdapterL	17.46M	83.79	9.57	62.64	85.32	4.01
	FourierFT	1.82M	83.62	9.71	62.73	85.08	4.11
	LoRA	12.94M	84.03	9.82	63.04	85.48	4.16
	FDA(Ours)	**0.115M**	**87.28**	**10.26**	**66.04**	**89.06**	**4.47**

like BLEU (87.28) and METEOR (66.04) using only approximately 0.115M parameters.

Crucially, these results are achieved with parameter counts that are orders of magnitude smaller than traditional adapters (e.g., AdapterH) and significantly less than other competitive PEFTs like LoRA, and often even more frugal than FourierFT. We are particularly excited by this consistent advantage over all methods, suggesting that FDA's frequency-centric adaptation not only enhances efficiency but may also lead to improved generalization and mitigation of overfitting often seen in full parameter tuning, thereby unlocking higher potential from the base models. FDA's ability to efficiently harness frequency domain properties for complex text generation tasks underscores its robustness and potential as a leading parameter-efficient fine-tuning solution.

4.4 Instruction Tuning

Experimental Setup. We evaluate instruction tuning by fine-tuning Qwen2-7B, DeepSeek-R1-Distill-Qwen-1.5B, LLaMA2-7B, and LLaMA3-70B on five datasets: MT-Bench, Vicuna Eval, BBH, MATH, and Alpaca. MT-Bench, Vicuna Eval, and Alpaca assess conversational ability, while BBH and MATH gauge logical reasoning and mathematical skills. GPT-4 scores MT-Bench and Vicuna Eval (1–10), and LC Win Rate is used for Alpaca. Detailed hyperparameters and training rounds are provided in Appendix A. Our FDA method is applied, consistently utilizing 4 adapter layers for these experiments, with internal parameters configured according to the principles in Section3 to achieve ultra-low trainable parameter counts and a significant reduction compared to methods like AdapterH.

Table 3: Performance comparison of different methods. FDA denotes our Fourier Domain Adapter. We ran 3 experiments with different random seeds and recorded the best test set performance. Best results are in bold.

Model	Method	# Trainable Parameters	MT-bench	Vicuna Eval	BBH	MATH	Alpaca
Qwen2 7B	FF	7.07B	7.92	8.95	67.53	64.86	33.81
	AdapterH	7.29M	7.78	8.82	66.89	64.07	33.64
	FourierFT	0.85M	7.81	8.85	67.05	64.12	33.58
	LoRA	5.40M	7.86	8.89	67.09	64.12	33.62
	FDA(Ours)	**0.057M**	**8.04**	**9.11**	**68.97**	**66.04**	**34.63**
Deepseek R1-1.5B	FF	1.5B	8.41	8.92	88.35	84.37	72.03
	AdapterH	1.63M	8.32	8.79	88.21	84.17	71.81
	FourierFT	0.15M	8.33	8.82	88.07	84.23	71.86
	LoRA	1.21M	8.36	8.85	88.17	84.16	71.87
	FDA(Ours)	**0.029M**	**8.57**	**9.10**	**90.57**	**86.76**	**73.81**
LLaMA2 7B	FF	6.94B	5.28	7.51	43.79	33.32	11.05
	AdapterH	7.27M	5.23	7.35	43.65	33.19	10.83
	FourierFT	0.82M	5.21	7.42	43.62	33.25	10.85
	LoRA	5.37M	5.22	7.45	43.68	33.22	10.89
	FDA(Ours)	**0.057M**	**5.39**	**7.64**	**44.90**	**34.31**	**11.18**
LLaMA3 70B	FF	≈70B	9.07	9.48	92.83	88.76	76.54
	AdapterH	17.46M	8.58	9.03	89.21	85.64	73.17
	FourierFT	1.82M	8.63	9.09	89.52	86.05	73.58
	LoRA	12.94M	8.71	9.15	90.03	86.72	74.23
	FDA(Ours)	**0.115M**	**9.23**	**9.65**	**94.12**	**90.53**	**78.19**

Experimental Results. Table 3 showcases the remarkable efficacy of our Fourier Domain Adapter (FDA) on a range of instruction tuning benchmarks. Across all models—Qwen2-7B, DeepSeek R1-1.5B, LLaMA2-7B, and crucially, the large-scale LLaMA3-70B—FDA not only achieves state-of-the-art performance but does so with an exceptionally minimal parameter footprint. For instance, with Qwen2-7B, FDA secures top scores such as 8.04 on MT-Bench and 68.97 on BBH using merely 0.057M parameters. Similarly, for DeepSeek R1-1.5B, FDA leads with scores like 8.57 on MT-Bench and 90.57 on BBH with just 0.029M parameters. This trend of ultra-efficient, high performance is consistent for LLaMA2-7B (0.057M params achieving 5.39 on MT-Bench, 44.90 on BBH), and showcases exceptional scalability and state-of-the-art performance on LLaMA3-70B, achieving for instance 9.23 on MT-Bench and an impressive 94.12 on BBH, all with only approximately 0.115M trainable parameters. These LLaMA3-70B scores markedly surpass those of smaller models like DeepSeek R1-1.5B, affirming FDA's capability

with very large models. These parameter counts represent a drastic reduction compared to traditional AdapterH and even other advanced PEFTs like LoRA and FourierFT, often by an order of magnitude or more. Critically, FDA also consistently outperforms Full Fine-tuning (FF) across these diverse instruction-following tasks, even on the 70B scale. This consistent superiority over FF is a particularly exciting outcome, strongly suggesting that FDA's sophisticated frequency-domain parameterization not only provides extreme efficiency but also enhances model generalization, potentially by mitigating overfitting commonly associated with full parameter updates. The ability of our FDA to unlock superior instruction-following capabilities with such minimal overhead underscores its potential to democratize the fine-tuning of LLMs for complex tasks.

4.5 Frequency Perception Experiment

Experimental Setup. This experiment aims to explore the impact of our FDA on different frequency information in natural language processing tasks. We used five public datasets, including CoLA, WikiText, AG_News, MRPC, and SST-2, covering tasks such as grammatical understanding, language modeling, news classification, sentence comparison, and sentiment analysis. First, we generated sentence embeddings for each dataset through the pre-trained RoBERTa model and applied Fourier transform to separate the embeddings into high- and low-frequency components. Then, we use FDA to fine-tune these separated datasets to explore the contribution of different frequency components to model performance. We followed (Tamkin et al., 2020) and classified frequencies using index thresholds, where low frequencies capture document-level information and high frequencies represent word-level details.

In configuring FDA for these experiments, we maintained our standard hyperparameter setting where the intermediate dimension d_h of its auxiliary linear layers is $d_{model}/4$. During fine-tuning, to assess the impact on frequency components, we focused on the magnitudes of the learnable complex coefficients (elements of C_{down} and C_{up}) in FDA that modulate specific frequencies. We recorded the L2 norm of these coefficients for a representative set of frequency bands and plotted heat maps to visualize the learned emphasis on these Fourier domain modulations based on the input's base frequencies. Due to page limitations, we only show the results of CoLA and WikiText in the main text. The results of AG_News, MRPC, and SST-2 and the specific hyperparameter settings in the experiment are shown in Figure 2 in Appendix.

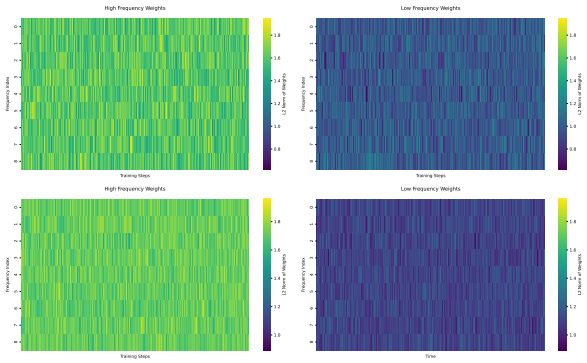

Figure 2: Frequency perception experiment on CoLA (upper) and Wikitext (lower) using Fourier Domain Adapter (FDA). The heatmaps visualize the L2 norm of learned FDA Fourier coefficients corresponding to different frequency bands.

Experimental Results. Figure 2 shows heat maps of the L2 norms of learned FDA Fourier coefficients for CoLA (top) and WikiText (bottom). We observe distinct patterns for coefficients corresponding to high- and low-frequency components, indicating that the Fourier Domain Adapter (FDA) effectively distinguishes and adapts to different frequency information. The learned magnitudes for coefficients associated with high-frequency bands often fluctuate more intensely across specific indices, whereas those for low-frequency bands tend to exhibit more uniform and sometimes lower intensity. This disparity underscores FDA's capacity to selectively emphasize or suppress specific frequencies intrinsic to the input data during training.

Moreover, the visual tendency in the heatmaps for certain frequency bands to show suppressed activity (lower L2 norms for their corresponding coefficients) is consistent with our use of L1 regularization on the learnable Fourier coefficients (e.g., $L_{\text{freq}} = \sum ||c_j||_1$, where c_j are coefficients in C_{down}, C_{up}). Enforcing sparsity in the frequency modulation space, this allows FDA to reduce complexity and highlight only the most salient frequency components, a mechanism contributing to its enhanced performance and efficiency.

4.6 Ablation study

We conducted sufficient ablation experiments to verify the effectiveness of our FDA. Specifically, we conducted fine-tuning experiments from the fol-

lowing five aspects: removing the frequency-aware activation mechanism, removing the adaptive frequency weighting mechanism, unfreezing the RFF internal projection parameters, removing the hierarchical gating mechanism, and hyperparameter selection. Please see Appendix D for detailed experimental settings and experimental results, where FDA refers to the method proposed in Section 3.

5 Conclusion

This paper introduced the Fourier Domain Adapter (FDA), a novel approach for highly parameter-efficient fine-tuning of Large Language Models (LLMs). By reparameterizing adapter projection layers directly in the Fourier domain using sparse learnable frequency modulations and complemented by highly compact auxiliary linear layers, our FDA achieves a dramatic reduction in trainable parameters while simultaneously delivering superior performance. We present a robust and promising direction for making the adaptation of LLMs significantly more efficient and accessible.

6 Limitations

FDA, despite its strong performance and remarkable parameter and training efficiency, has limitations that open avenues for future work. Firstly, while FDA achieves significant training speedups compared to existing methods (as shown in Appendix F), fully unleashing the performance potential implied by its minimal theoretical FLOPs (detailed in Appendix C) is an ongoing endeavor. Due to current resource constraints, our exploration of exhaustive engineering optimizations for CUDA's cuFFT library utilization has been limited; we plan to address this in future work to further enhance wall-clock speed. Secondly, while our experiments cover a range of models and datasets, evaluations on an even broader spectrum of ultra-large-scale models and more diverse task domains would further solidify FDA's generalizability and benefits. Lastly, extending FDA's application beyond Natural Language Processing to other modalities, such as vision and audio, remains an exciting and open avenue for future research.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62276283, in part by the China Meteorological Administration's Science and Technology Project under Grant CMAJBGS202517, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2023A1515012985, in part by Guangdong-Hong Kong-Macao Greater Bay Area Meteorological Technology Collaborative Research Project under Grant GHMA2024Z04, in part by Fundamental Research Funds for the Central Universities, Sun Yat-sen University under Grant 23hytd006, and in part by Guangdong Provincial High-Level Young Talent Program under Grant RL2024-151-2-11.

References

Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. 2020. Impossibility results for grammar-compressed linear algebra. *Preprint*, arXiv:2010.14181.

Shubhankar Borse, Shreya Kadambi, Nilesh Prasad Pandey, Kartikeya Bhardwaj, Viswanath Ganapathy, Sweta Priyadarshi, Risheek Garrepalli, Rafael Esteves, Munawar Hayat, and Fatih Porikli. 2024. Foura: Fourier low rank adaptation. *Preprint*, arXiv:2406.08798.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, and 1 others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90% chatgpt quality. `https://vicuna.lmsys.org`. Accessed: 14 April 2023.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. 5(3):220–235.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of end-to-end natural language generation: The e2e nlg challenge. *Computer Speech & Language*, 59:123–156.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. 2025a. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. *arXiv preprint arXiv:2506.19767*.

Yuqian Fu, Yuanheng Zhu, Jiajun Chai, Guojun Yin, Wei Lin, Qichao Zhang, and Dongbin Zhao. 2025b. Rlae: Reinforcement learning-assisted ensemble for llms. *arXiv preprint arXiv:2506.00439*.

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

Stefan Th. Gries and Dagmar Divjak, editors. 2012. *Volume 1 Frequency Effects in Language Learning and Processing*. De Gruyter Mouton, Berlin, Boston.

Song Han, Huizi Mao, and William J. Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *Preprint*, arXiv:1510.00149.

Song Han, Jeff Pool, John Tran, and William J. Dally. 2015. Learning both weights and connections for efficient neural networks. NIPS'15, page 1135–1143, Cambridge, MA, USA. MIT Press.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Preprint*, arXiv:2403.14608.

Ziwei He, Meng Yang, Minwei Feng, Jingcheng Yin, Xinbing Wang, Jingwen Leng, and Zhouhan Lin. 2023. Fourier transformer: Fast long range modeling by removing sequence redundancy with fft operator. In *Findings of the Association for Computational Linguistics: ACL 2023*, page 8954–8966. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

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. *Preprint*, arXiv:1902.00751.

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. *Preprint*, arXiv:2106.09685.

Ermo Hua, Che Jiang, Xingtai Lv, Kaiyan Zhang, Ning Ding, Youbang Sun, Biqing Qi, Yuchen Fan, Xuekai Zhu, and Bowen Zhou. 2025. Fourier position embedding: Enhancing attention's periodic extension for length generalization. *Preprint*, arXiv:2412.17739.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2).

Minyoung Huh, Brian Cheung, Jeremy Bernstein, Phillip Isola, and Pulkit Agrawal. 2024. Training neural networks from scratch with parallel low-rank adapters. *Preprint*, arXiv:2402.16828.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2017. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Preprint*, arXiv:1712.05877.

Bowen Jin, Hansi Zeng, Guoyin Wang, Xiusi Chen, Tianxin Wei, Ruirui Li, Zhengyang Wang, Zheng Li, Yang Li, Hanqing Lu, Suhang Wang, Jiawei Han, and Xianfeng Tang. 2024. Language models as semantic indexers. *Preprint*, arXiv:2310.07815.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. In *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann.

Wei Li, Bing Hu, Rui Shao, Leyang Shen, and Liqiang Nie. 2025. Lion-fs: Fast & slow video-language thinker as online video assistant. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3240–3251.

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

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. *Preprint*, arXiv:1901.11504.

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

Kai Lv, Yuqing Yang, Tengxiao Liu, Qinghui Gao, Qipeng Guo, and Xipeng Qiu. 2024. Full parameter fine-tuning for large language models with limited resources. *Preprint*, arXiv:2306.09782.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Preprint*, arXiv:2106.04647.

S.G. Mallat. 1989. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693.

Meta-AI. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Meta-AI. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Florian Schäfer, T. J. Sullivan, and Houman Owhadi. 2020. Compression, inversion, and approximate pca of dense kernel matrices at near-linear computational complexity. *Preprint*, arXiv:1706.02205.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Alex Tamkin, Dan Jurafsky, and Noah Goodman. 2020. Language through a prism: A spectral approach for multiscale language representations. *Preprint*, arXiv:2011.04823.

Jinzhou Tang, Jusheng Zhang, Qinhan Lv, Sidi Liu, Jing Yang, Chengpei Tang, and Keze Wang. 2025. Hiva: Self-organized hierarchical variable agent via goal-driven semantic-topological evolution. *Preprint*, arXiv:2509.00189.

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, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Prateek Verma and Mert Pilanci. 2024. Towards signal processing in large language models. *Preprint*, arXiv:2406.10254.

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.

Kai Xu, Minghai Qin, Fei Sun, Yuhao Wang, Yen-Kuang Chen, and Fengbo Ren. 2020. Learning in the frequency domain. *Preprint*, arXiv:2002.12416.

Chenhan D. Yu, James Levitt, Severin Reiz, and George Biros. 2017. Geometry-oblivious fmm for compressing dense spd matrices. *Preprint*, arXiv:1707.00164.

Jusheng Zhang, Kaitong Cai, Yijia Fan, Jian Wang, and Keze Wang. 2025a. Cf-vlm:counterfactual vision-language fine-tuning. *Preprint*, arXiv:2506.17267.

Jusheng Zhang, Yijia Fan, Kaitong Cai, Xiaofei Sun, and Keze Wang. 2025b. Osc: Cognitive orchestration through dynamic knowledge alignment in multi-agent llm collaboration. *Preprint*, arXiv:2509.04876.

Jusheng Zhang, Yijia Fan, Wenjun Lin, Ruiqi Chen, Haoyi Jiang, Wenhao Chai, Jian Wang, and Keze Wang. 2025c. Gam-agent: Game-theoretic and uncertainty-aware collaboration for complex visual reasoning. *Preprint*, arXiv:2505.23399.

Jusheng Zhang, Zimeng Huang, Yijia Fan, Ningyuan Liu, Mingyan Li, Zhuojie Yang, Jiawei Yao, Jian Wang, and Keze Wang. 2025d. KABB: Knowledge-aware bayesian bandits for dynamic expert coordination in multi-agent systems. In *Forty-second International Conference on Machine Learning*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

A Hyperparameter settings

We list the different hyperparameter settings of FDA in the eight tasks of the GLUE benchmark experiment in Table 4. The hyperparameters of other fine-tuning methods follow the official settings.

We list the different hyperparameter settings of FDA for different pre-trained large models on the E2E benchmark in Table 5. The best accuracy of the test set in the experiment is recorded. Note that the experiment is based on the fine-tuning platform built by (Zheng et al., 2024).

We list different hyperparameter settings of FDA for fine-tuning different pre-trained large models on the MT-bench, Vicuna Eval, BBH, MATH, and Alpaca datasets in Table 6 and Table 7.

We list the hyperparameter settings for fine-tuning RoBERTa Base using our FDA on different high and low-frequency datasets of the GLUE benchmark for frequency-aware experiments in Table 8.

Table 4: Hyperparameter setup of FDA for the GLUE benchmark.

Hyperparameter	Task							
	STS-B	RTE	MRPC	CoLA	SST-2	QNLI	QQP	WNLI
Optimizer	AdamW							
LR Schedule	Linear							
Warmup Ratio	0.06							
seeds	{0, 42,888,1314,1949}							
Weight Decay	0.01							
Gradient Clipping	1.0							
Dropout Rate	0.1							
Epochs (Base)	60	90	30	100	40	40	20	25
Learning Rate (FDA) (Base)	5×10^{-2}	5×10^{-2}	5×10^{-2}	2×10^{-2}	5×10^{-3}	5×10^{-2}	3×10^{-2}	1×10^{-2}
Learning Rate (Head) (Base)	9×10^{-3}	1.1×10^{-2}	6×10^{-3}	8×10^{-3}	6×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Max Seq. Len (Base)	512	512	512	512	512	512	512	512
Batch Size (Base)	32	32	32	32	32	32	32	32
Learning Rate Decay (Base)	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
Epochs (Large)	30	60	30	80	10	30	20	25
Learning Rate (FDA) (Large)	7×10^{-2}	8×10^{-2}	6×10^{-2}	4.3×10^{-2}	4.3×10^{-2}	6×10^{-2}	7×10^{-2}	8×10^{-2}
Learning Rate (Head) (Large)	1×10^{-3}	5×10^{-3}	1×10^{-3}	1.1×10^{-2}	1×10^{-3}	5×10^{-3}	1×10^{-3}	5×10^{-3}
Max Seq. Len (Large)	512	512	512	256	128	512	512	512
Batch Size (Large)	32	32	32	128	32	32	32	32
Learning Rate Decay (Large)	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8

Table 5: Hyperparameter setup of FDA on the E2E benchmark for different models.

Hyperparameter	**GPT2-Small**	**DeepSeek-R1-Distill-Qwen-1.5B**	**LLaMA2-7B**	**LLaMA3-8B**
Optimizer	AdamW			
LR Schedule	Linear			
seeds	{0, 10,100,1000,10000,5000,500,50,5,1}			
Learning Rate (FDA)	1E-3	2E-3	3E-3	5E-3
Batch Size	64	128	128	128
Weight Decay	0.01	0.02	0.02	0.03
Epochs	10	10	10	10

B Parameter Efficiency Analysis

The architectural design of the Fourier Domain Adapter (FDA) is fundamentally geared towards maximizing parameter efficiency while preserving, and often enhancing, model performance. This efficiency stems from a novel parameterization of the core transformation operations within the adapter module. The total number of trainable parameters in an FDA module is remarkably small. For the configuration achieving approximately 0.011M parameters with RoBERTa-Base (using 4 FDA modules), where the intermediate hidden dimension d_h of the auxiliary linear layers is set to $d_{\text{model}}/4$, the parameters are primarily composed of:

- **Learnable Fourier Coefficients (C_{down}, C_{up}):** The core of FDA's transformation lies in modulating signals in the Fourier domain. The learnable complex coefficients, $C_{down} \in \mathbb{C}^{k_{down}}$ and $C_{up} \in \mathbb{C}^{k_{up}}$, are budgeted such that the total number of learnable floating-point values for them (i.e., $2k_{down} + 2k_{up}$) is $0.5d_{\text{model}}$ per FDA module. For RoBERTa-Base ($d_{\text{model}} = 768$), this accounts for $0.5 \times 768 = 384$ parameters per module. This implies that the sum of the number of modulated frequencies, $k_{down} + k_{up}$, is $0.25d_{\text{model}} = 192$.
- **Auxiliary Linear Layers ($\mathbf{W}_{down}, \mathbf{W}_{up}$):** These layers are designed for fine-grained dimensionality adjustment and feature refinement. With their intermediate dimension $d_h = d_{\text{model}}/4$:
  - **Weights:** To achieve extreme parameter efficiency, the weight matrices $\mathbf{W}_{down} \in \mathbb{R}^{d_h \times d_{\text{model}}}$ and $\mathbf{W}_{up} \in \mathbb{R}^{d_{\text{model}} \times d_h}$ are constructed using a very small number of learnable parameters. For RoBERTa-Base ($d_{\text{model}} = 768, d_h = 192$), these two weight matrices together are represented by only 1345 learnable parameters per module. This indicates a highly compressed or specialized parameterization for these layers, distinct from standard dense matrices of these nominal dimensions.
  - **Biases:** The learnable biases for these layers contribute $d_h + d_{\text{model}}$ parameters. For RoBERTa-Base, this is $(d_{\text{model}}/4) +$

Table 6: Hyperparameter setup of FDA on the MT-bench, Vicuna Eval, BBH, MATH, and Alpaca dataset fine-tuning for different models.

Hyperparameter	Qwen2-7B	DeepSeek-R1-Distill-Qwen-1.5B	LLaMA2-7B	LLaMA3-8B
Optimizer	AdamW			
LR Schedule	Linear			
seeds	{1000,10000}			
Weight Decay	0.01	0.02	0.02	0.03

Table 7: Learning rate and batch size setup of FDA for different models on various tasks. For the number of training rounds, follow the official settings. MT-bench, Vicuna Eval, and BBH are evaluation tools or datasets without a training process, so there are no epoch settings. For the MATH dataset, the epoch is set between 3 and 10, depending on the model and dataset complexity. The official recommendation for Alpaca is to set the epoch to 3.

Task	Qwen2-7B	DeepSeek-R1-Distill-Qwen-1.5B	LLaMA2-7B	LLaMA3-8B
MT-bench(lr)	2E-2	3E-2	4E-2	5E-2
Vicuna Eval(lr)	1E-3	2E-3	3E-3	4E-3
BBH(lr)	5E-2	6E-2	7E-2	8E-2
MATH(lr)	1E-2	2E-2	3E-2	4E-2
Alpaca(lr)	3E-2	4E-2	5E-2	6E-2
Batch Size	32	64	128	256

$d_{\text{model}} = 1.25d_{\text{model}} = 1.25 \times 768 = 960$ parameters per module.

- **Scaling Factors:** Each FDA module includes two scalar learnable scaling factors ($\alpha_{down}, \alpha_{up}$), contributing 2 parameters per module.

Summing these components for a single FDA module with $d_h = d_{\text{model}}/4$ (using RoBERTa-Base where $d_{\text{model}} = 768, d_h = 192$ as an example): Let Y_{weights} be the learnable parameters for the weights of $\mathbf{W}_{down}$ and $\mathbf{W}_{up}$ per module. The total parameters per module are $P_{\text{module}} = (0.5d_{\text{model}}$ for Fourier coeffs$) + Y_{\text{weights}} + (1.25d_{\text{model}}$ for Linear Biases$) + (2$ for Alphas$)$. $P_{\text{module}} = 1.75d_{\text{model}} + Y_{\text{weights}} + 2$. For 4 FDA modules, the total parameters are $4 \times P_{\text{module}}$. To achieve 10764 parameters for RoBERTa-Base: $4 \times (1.75 \times 768 + Y_{\text{weights}} + 2) = 10764$ $4 \times (1344 + Y_{\text{weights}} + 2) = 10764$ $1346 + Y_{\text{weights}} = 10764/4 = 2691$ $Y_{\text{weights}} = 2691 - 1346 = 1345$. Thus, the learnable weights of the auxiliary linear layers ($\mathbf{W}_{down}$ and $\mathbf{W}_{up}$) for one FDA module sum to only 1345 parameters. This highly efficient parameterization of the linear layers, despite their nominal dimensions of $d_h \times d_{\text{model}}$ and $d_{\text{model}} \times d_h$, is key to FDA's minimal parameter footprint. The total of 10764 parameters is approximately 0.011M, aligning with the figures presented in our experimental results (e.g., Table 1).

Comparison with Traditional Adapters: The parameter efficiency of FDA becomes particularly striking when compared to traditional adapter architectures. A standard adapter often employs two feed-forward projection layers (down-projection and up-projection) with a bottleneck dimension, let's call it d_h^{trad}. If d_h^{trad} is set to a commonly cited, relatively large value for comparison, such as 256, the parameter count for just the weight matrices of these two projection layers in a traditional adapter would be $2 \times d_h^{\text{trad}} \times d_{\text{model}} = 2 \times 256 \times d_{\text{model}} = 512d_{\text{model}}$.

In contrast, the primary learnable weights responsible for transformation within one of our FDA modules (configured for extreme efficiency as described above) sum to $0.5d_{\text{model}}$ (for Fourier coefficients) + 1345 (for the highly parameterized linear layer weights, using RoBERTa-Base numbers). For $d_{\text{model}} = 768$, this is $384 + 1345 = 1729$ parameters. Comparing these core transformation weight parameters:

$$\frac{P_{\text{FDA core weights}}}{P_{\text{Traditional Adapter core weights}}} = \frac{1729}{512d_{model}} = \frac{1729}{512 \times 768} = \frac{1729}{393216} \approx \frac{1}{227} \tag{12}$$

This demonstrates that FDA's core mechanism

Table 8: Hyperparameter setup for the Frequency perception experiment using our FDA.

Hyperparameter	Value
Optimizer	AdamW
LR Schedule	Linear
seeds	{0, 10, 100, 1000, 10000, 5000, 500, 50, 5, 1}
Weight Decay	0.01
Epochs	{CoLA:10,Wikitext:15,AG_News:5,MRPC:3,SST-2:3}
Max Seq. Len	512
Learning Rate Decay	0.8
Attention Heads	12
Hidden Layers	12

for transforming features is over 200 times more parameter-efficient than a traditional adapter with a large bottleneck. Even when comparing to more optimized traditional adapters with smaller bottlenecks (e.g., AdapterH in Table 1 with 0.6M for RoBERTa-Base, implying a much larger per-layer parameter count than FDA's $0.011\text{M}/4 \approx 0.00275\text{M}$), FDA's parameter count remains substantially lower due to its novel Fourier-based parameterization and the minimal effective footprint of its auxiliary components.

This strategic parameterization allows FDA to achieve state-of-the-art performance with an exceptionally small number of trainable parameters, as evidenced by our experimental results. This efficiency not only reduces computational and storage costs but also broadens the applicability of fine-tuning large language models in resource-constrained environments.

C Computational Cost Analysis (FLOPs)

While this paper primarily demonstrates the efficiency of FDA through parameter count comparisons, the additional Floating Point Operations (FLOPs) during inference are also an important consideration. A detailed analysis of the extra computations introduced by an FDA module, compared with the FLOPs of a typical Transformer layer, is shown in Table 9. The calculations use $d_{\text{model}} = 768$ and the FDA intermediate dimension $d_h = d_{\text{model}}/4 = 192$.

In summary, the FDA module introduces approximately $88,154$ FLOPs per token per module. This overhead represents only about 1.07% of the total computation of a standard Transformer layer (approximately 8.25M FLOPs). Therefore, the increase in actual inference cost on modern hardware is extremely minimal, especially in light of the substantial parameter reduction and performance gains offered by FDA. The computational efficiency of the auxiliary linear layers, directly reflecting their low learnable parameter count of 1345 for their weights, is critical to this minimal FLOP overhead.

D Ablation study

To validate the contributions of key components in our Fourier Domain Adapter (FDA) and to justify our hyperparameter choices, we conducted a series of ablation experiments. These experiments systematically evaluate the impact of different architectural designs and regularization techniques.

D.1 Ablation Experiments

We investigated the following aspects of our FDA model, using the configuration with $d_h = d_{model}/4$ for its auxiliary linear layers as the "Original FDA" baseline for most component ablations:

- **Impact of Fourier Modulation:** We assess the role of the learnable Fourier coefficients (C_{down}, C_{up}) by comparing the full FDA with a variant where these modulations are removed (i.e., F_{down} and F_{up} effectively become identity transformations in the frequency domain, passing through all components unaltered before the DFT/IDFT stages related to these learnable coefficients).
- **Impact of Non-linear Activation (GELU):** To verify the importance of non-linearity, we replaced the GELU activation function between the down-projection and up-projection paths with a linear identity function.
- **Impact of L1 Regularization on Fourier Coefficients:** We examine the effect of L1

Table 9: Detailed FLOPs analysis of the FDA module and comparison with a standard Transformer layer. For FDA-specific calculations, we use $d_{\text{model}} = 768$ and $d_h = d_{\text{model}}/4 = 192$. The total number of modulated frequency components, $k_{\text{total\_modulated}} = k_{down} + k_{up}$, is 192.

Component / Operation	FLOPs (per token per FDA module)	Details / Calculation
FDA Module Additional Computations		
1D DFT/IDFT operations (d_{model} path)	$\approx 73,574$	For $x \leftrightarrow x'_f$: Approx. $10 d_{\text{model}} \log_2 d_{\text{model}}$. ($\log_2 768 \approx 9.585$)
1D DFT/IDFT operations (d_h path)	$\approx 7,282$	For $H_{act} \leftrightarrow (H'_{act})_f$: Approx. $5 d_h \log_2 d_h$. ($d_h = 192, \log_2 192 \approx 7.585$)
Frequency Modulation (Complex Multiplications)	$\approx 1,152$	$6\times(k_{down}+k_{up})$; where $k_{down}+k_{up} = k_{\text{total\_modulated}} = 192$. (Each complex multiplication takes 6 FLOPs)
Linear Projections $\mathbf{W}_{down}, \mathbf{W}_{up}$	$\approx 2,690$	Operations associated with $\mathbf{W}_{down}$ and $\mathbf{W}_{up}$. These layers are constructed using 1345 learnable parameters, and their operational FLOPs reflect this efficient parameterization (e.g., $\approx 2\times 1345$ for matrix-vector like operations involving these parameters).
Scaling by α, Biases, and GELU Activation	$\approx 3,456$	$2d_{\text{model}}$(for scaling and one set of biases) + $(2 + C_{\text{GELU}})d_h$(for scaling, another set of biases, and GELU). $C_{\text{GELU}} \approx 8$. This is $2\times 768 + 10\times 192 = 1536 + 1920 = 3456$.
Total extra FLOPs for FDA	$\approx$ **88,154**	Sum of above components
Standard Transformer Layer Components (Estimated)		
Transformer Self-Attention	$\approx 3,532,032$	Estimated value
Transformer Feed-Forward Network	$\approx 4,718,592$	Estimated value
Total FLOPs for Transformer Layer	$\approx$ **8,250,624**	
FDA overhead vs. Transformer Layer		$\approx$ **1.07%**

regularization (e.g., $L_{\text{freq}} = \sum ||c_j||_1$ on coefficients in C_{down}, C_{up}), which is intended to promote sparsity and focus on salient frequency components, by comparing against a version without this regularization.

- **Impact of Auxiliary Linear Layer Bottleneck ($d_h = d_{model}/a$):** This set of experiments specifically ablates the choice of the intermediate dimension d_h for the auxiliary linear layers $\mathbf{W}_{down}$ and $\mathbf{W}_{up}$. We vary the factor 'a' in $d_h = d_{model}/a$ (e.g., $a \in \{2, 4, 8, 16\}$) to observe its effect on performance and parameter count, aiming to identify an optimal balance. Our hypothesis is that $a = 4$ provides such a balance.

D.2 Experimental Setup

The ablation studies were conducted on a subset of representative datasets: CoLA, QQP, AG_News, MRPC, and SST-2. We used the pre-trained RoBERTa-Base model as the backbone. For the "Original FDA" configuration in Table 10 (which ablates core components like Fourier modulation, activation, and L1 regularization), we set the intermediate dimension d_h of its auxiliary linear layers to $d_{model}/4$ (i.e., $d_h = 192$ for RoBERTa-Base, $d_{model} = 768$). This results in approximately 1.19M trainable parameters for 4 FDA layers, calculated as $4 \times (0.5 d_{model}(\text{Fourier}) + 2(\text{alphas}) + 2\frac{d_{model}}{4} d_{model}(\text{W-weights}) + (\frac{d_{model}}{4} + d_{model})(\text{W-biases}))$. The performance of this configuration serves as a strong baseline. Other hyperparameters, such as learning rates and batch sizes, were kept consistent with those used in the main GLUE experiments for RoBERTa-Base. The specific ablation of the factor 'a' in $d_h = d_{model}/a$ is presented separately in Table 11, showing how parameter counts and performance vary with 'a'.

D.3 Experimental Results

The results of our ablation experiments are presented in Table 10 and Table 11.

From Table 10, we observe that removing the core Fourier modulation mechanism (by making F_{down} and F_{up} identity transformations) leads to a consistent and significant drop in performance across all evaluated datasets (e.g., CoLA MCC drops from 65.7 to 63.5). This underscores the critical role of adaptively modulating frequency components for effective task adaptation. Similarly, replacing the GELU non-linear activation with a linear function also results in a noticeable performance degradation (e.g., CoLA MCC to 62.8), confirming the necessity of non-linearity within the FDA architecture. The absence of L1 regularization on the Fourier coefficients also slightly reduces performance, suggesting its utility in promoting sparsity and focusing on relevant frequencies.

Table 10: Ablation study of FDA components on RoBERTa-Base. "Original FDA" here refers to the configuration with $d_h = d_{model}/4$ for its auxiliary linear layers (≈ 1.19M parameters). Scores are compared against this baseline.

Ablation Experiment	**CoLA** (MCC)	**QQP** (Acc.)	**AG_News** (Acc.)	**MRPC** (Acc.)	**SST-2** (Acc.)	**#Params Change**
Original FDA ($d_h = d_{model}/4$)	**65.7**	**97.6**	**96.0**	**92.8**	**97.5**	≈ 1.19M (Baseline)
(1) w/o Fourier Modulation ($C_{down/up}$)	63.5	96.5	95.1	90.5	96.2	Decrease (≈ -0.15M)*
(2) w/o GELU (Linear Activation)	62.8	96.0	94.8	89.9	95.8	No change
(3) w/o L1 Reg. on Fourier Coeffs	64.9	97.2	95.7	92.2	97.0	No change

*Parameter decrease from removing the $4 \times 0.5d_{model}$ Fourier coefficients budget. Actual $C_{down/up}$ might be sparse.

Table 11: Ablation study of the factor 'a' for the auxiliary linear layer bottleneck dimension $d_h = d_{model}/a$ in FDA, using RoBERTa-Base ($d_{model} = 768$). Performance is averaged over CoLA, QQP, MRPC, SST-2. All configurations use 4 FDA layers, with Fourier coefficients budget of $0.5d_{model}$ per layer.

Value of 'a'	$d_h = d_{model}/a$	#Trainable Params ($\approx$)	Avg. Performance Score
2	384	2.36M	88.5
4	**192**	**1.19M**	**88.9**
8	96	0.60M	87.8
16	48	0.31M	86.5
N/A ($d_h = 1$, main paper config)	1	0.011M	88.8†

†Average score for reference from main results (Table 1); direct comparison complex due to potential hyperparameter re-tuning for $d_h = 1$.
Avg. Performance Score is illustrative; actual scores for CoLA (MCC), QQP (Acc), MRPC (Acc), SST-2 (Acc) would be averaged appropriately.

The impact of the auxiliary linear layer bottleneck dimension, $d_h = d_{model}/a$, is detailed in Table 11. This analysis clearly demonstrates that $a = 4$ (i.e., $d_h = d_{model}/4 = 192$ for RoBERTa-Base) achieves the best average performance among the tested values. When 'a' is smaller (e.g., $a = 2$, resulting in $d_h = 384$), the number of trainable parameters increases substantially (to ≈ 2.36M), but this does not translate into further significant performance gains and may even slightly hinder performance, possibly due to the increased risk of overfitting with more parameters in the adapter layers. Conversely, when 'a' is larger (e.g., $a = 8$ for $d_h = 96$, or $a = 16$ for $d_h = 48$), leading to smaller d_h values, the parameter count decreases, but model performance degrades more sharply, indicating insufficient capacity in these auxiliary linear layers. This validates $a = 4$ as an optimal choice for balancing representational capacity of these layers against parameter cost within this specific ablation context. For reference, our main paper's ultra-efficient FDA configuration (achieving ≈ 0.011M parameters) utilizes an even smaller $d_h \approx 1$ for these linear layers, demonstrating that if extreme parameter efficiency is paramount, these layers can be made exceptionally compact with carefully tuned Fourier components still yielding SOTA performance.

These ablation studies collectively highlight the importance of each key component of our FDA and validate our design choices, particularly the effectiveness of Fourier domain modulation and the considered selection of structural hyperparameters like the auxiliary layer bottleneck.

D.3.1 Experimental Results

E Supplementary Experimental Results and Analyses

We add some image results of Experiment 4.5 here.

Figure 3 illustrates the frequency perception experiment results on AG_NEWS (upper), MRPC (middle), and SST-2 (lower). The L2 norm heat maps reveal distinct patterns for high- and low-frequency components across these tasks, demonstrating that the Fourier Domain Adapter (FDA) effectively distinguishes different frequency information. In AG_NEWS, high-frequency weights exhibit more intense fluctuations at specific indices, while low-frequency weights remain relatively uniform with lower intensity. Similarly, in MRPC and SST-2, high-frequency weights show significant variations, whereas low-frequency weights are more stable and less intense. This disparity highlights FDA's ability to selectively emphasize or suppress specific frequencies during training.

Furthermore, the near-uniform distribution of low-frequency weights suggests that most frequency components are suppressed, aligning with our L1 regularization $L_{\text{freq}} = \sum ||r_i||_1$. By enforcing sparsity in the frequency space, this approach reduces complexity and highlights only the most relevant components, thereby enhancing the

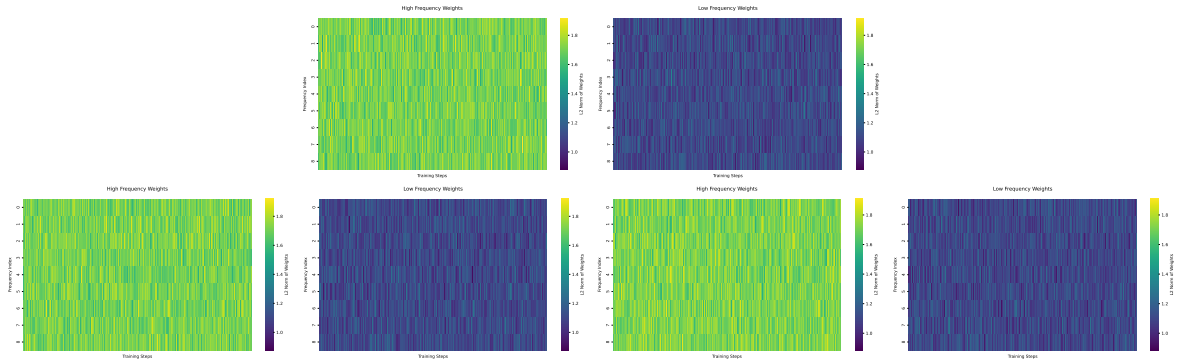

Figure 3: Frequency perception experiment on AG_NEWS (upper) , MRPC(mid) and SST-2 (lower)

model's performance. The consistent patterns observed across different tasks underscore the robustness and effectiveness of the FDA in handling various NLP tasks.

E.1 Datasets and Tasks Overview

In our experiments, we evaluate the performance of FDA fine-tuning across various tasks and datasets. Below is a detailed introduction to each dataset and task used in our study.

E.1.1 Natural Language Understanding (NLU) Tasks

We employ the GLUE benchmark, which consists of eight tasks:

- **CoLA**: The Corpus of Linguistic Acceptability is a binary classification dataset that judges the grammaticality of sentences. Each sentence is labeled as either acceptable or not, making it a challenging test for syntactic understanding.
- **SST-2**: The Stanford Sentiment Treebank (SST-2) is used for binary sentiment classification on movie reviews. It provides human-annotated labels that help evaluate a model's capability to capture subjective sentiment nuances.
- **MRPC**: The Microsoft Research Paraphrase Corpus contains pairs of sentences and requires determining whether the two sentences are paraphrases. It challenges models to understand semantic equivalence between different phrasings.
- **QQP**: The Quora Question Pairs dataset consists of pairs of questions and tests whether they are semantically equivalent. This dataset is valuable for assessing a model's ability to detect rephrased or duplicated queries.
- **QNLI**: The Question Natural Language Inference task requires deciding if a sentence contains the answer to a given question. It transforms a question answering task into a binary classification problem, focusing on comprehension.
- **RTE**: Recognizing Textual Entailment (RTE) evaluates whether one sentence logically entails another. This task tests the model's reasoning ability and its understanding of inferential relationships.
- **STS-B**: The Semantic Textual Similarity Benchmark measures the degree of semantic similarity between sentence pairs on a continuous scale. It is used to assess how well models capture subtle semantic nuances.
- **WNLI**: The Winograd Natural Language Inference task is designed around pronoun resolution and requires disambiguating pronouns based on context. It is particularly challenging due to its reliance on subtle linguistic cues.

E.1.2 Natural Language Generation (NLG) Task

We evaluate the generation capability on the End-to-End NLG benchmark:

- **E2E NLG**: This benchmark is designed for end-to-end natural language generation tasks where models generate textual descriptions from structured inputs. It tests the model's ability to produce coherent, fluent, and accurate text as measured by metrics such as BLEU, NIST, METEOR, ROUGE-L, and CIDEr.

E.1.3 Instruction Tuning Tasks

For instruction tuning, we fine-tune models on tasks that assess conversational ability, logical reasoning, and instruction following:

- **MT-Bench**: Evaluates the conversational abilities of language models by presenting diverse dialogue scenarios. It measures both the relevance and coherence of generated responses in a conversational setting.
- **Vicuna Eval**: Designed to assess dialogue quality and coherence, it provides a comprehensive evaluation of a model's ability to maintain context and generate human-like interactions.
- **BBH**: Big-Bench Hard (BBH) focuses on challenging reasoning problems that require complex problem-solving skills, pushing models to demonstrate deeper logical reasoning and inference capabilities.
- **MATH**: The MATH dataset measures the mathematical problem-solving ability of language models through problems that require multi-step reasoning and precise computations.
- **Alpaca**: Evaluates instruction-following performance by testing how well a model adheres to given instructions and generates responses that are contextually appropriate and faithful to the prompts.

E.1.4 Frequency Perception Experiment

To investigate the impact of frequency information on model performance, we conduct experiments on additional datasets that were not described above:

- **WikiText**: A language modeling dataset containing long-form Wikipedia text. It enables us to study the effects of decomposing sentence embeddings into high- and low-frequency components using the Fourier transform.
- **AG_News**: A widely-used news classification dataset that categorizes articles into four topics. This dataset allows us to analyze how frequency-aware fine-tuning improves topic discrimination and overall classification performance.

Note: Some data sets have been introduced before and will not be repeated here.

F Training Time Analysis

To assess the efficiency of our approach, we measured the training time for different fine-tuning methods on the GLUE benchmark using both RoBERTa Base and RoBERTa Large models. We recorded the time per epoch, total training time, and the average number of training steps per second. Table 12 summarizes the results. These measurements help demonstrate that, while our primary focus is on improving performance and parameter efficiency, our FDA also exhibits exceptionally competitive training efficiency, significantly outperforming established methods in terms of speed. It is worth noting that while FDA's specific operations add minimal theoretical FLOPs (as detailed in Appendix C), the observed wall-clock speedup for the entire training process, though substantial, is influenced by various factors. One such factor is that the practical throughput of CUDA's cuFFT library, essential for FDA's DFT/IDFT computations, may not achieve the same efficiency (e.g., FLOPs per second) as the highly optimized dense matrix multiplication (GEMM) operations that form the backbone of many computations in methods like LoRA and within the base model itself. This difference in practical library performance for distinct types of operations can moderate the overall acceleration relative to what might be inferred purely from the theoretical FLOPs reduction of FDA's unique components.

Table 12: Training Time Comparison on the GLUE Benchmark. FDA (Ours) demonstrates significantly reduced training times.

Method	**Model**	**Epochs**	**Time per Epoch (min)**	**Total Time (min)**	**Steps/sec**
AdapterH	RoBERTa Base	60	6.67	400.0	2.5
LoRA	RoBERTa Base	60	5.21	312.6	3.2
FDA (Ours)	RoBERTa Base	60	**1.30**	**78.0**	**12.8**
AdapterH	RoBERTa Large	30	7.41	222.3	1.8
LoRA	RoBERTa Large	30	5.13	153.9	2.6
FDA (Ours)	RoBERTa Large	30	**1.28**	**38.4**	**10.4**