Theoretical Insights into Fine-Tuning Attention Mechanism: Generalization and Optimization

Xinhao Yao[1,2,3*], **Hongjin Qian**[4], **Xiaolin Hu**[1], **Gengze Xu**[1], **Wei Liu**[5], **Jian Luan**[5], **Bin Wang**[5], **Yong Liu**[1,2,3†]
[1]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[2]Beijing Key Laboratory of Research on Large Models and Intelligent Governance
[3]Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE
[4]Beijing Academy of Artificial Intelligence
[5]XiaoMi
{yaoxinhao021978, liuyonggsai}@ruc.edu.cn

Abstract

Large Language Models (LLMs), built on Transformer architectures, exhibit remarkable generalization across a wide range of tasks. However, fine-tuning these models for specific tasks remains resource-intensive due to their extensive parameterization. In this paper, we explore two remarkable phenomena related to the attention mechanism during the fine-tuning of LLMs (where $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ denote the weights of the query, key, and value layers, respectively). The first phenomenon, termed *"Unequal Importance of Attention Matrices"*, highlights the impact of fine-tuning different weight matrices. It shows that optimizing the $\mathbf{W}_v$ matrix yields significantly better performance than optimizing the $\mathbf{W}_k$ matrix. Fine-tuning only the $\mathbf{W}_q$ and $\mathbf{W}_v$ matrices is computationally efficient while delivering results comparable to, or even better than fine-tuning all three matrices ($\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$). The second phenomenon, *"Attention Matrices with Customized Learning Rate Lead to Better Convergence"*, emphasizes the importance of assigning distinct learning rates to these matrices. Specifically, a higher learning rate for the $\mathbf{W}_v$ matrix compared to $\mathbf{W}_q$ and $\mathbf{W}_k$ accelerates convergence and improves performance. Building on these insights, we propose a new strategy that improves fine-tuning efficiency in terms of both storage and time. Experimental results on benchmark datasets validate the effectiveness of this approach, supporting our theoretical findings. Our analysis lays the theoretical groundwork for configuring and improving algorithms in LLMs fine-tuning.

1 Introduction

Large Language Models (LLMs) are often built on Transformer architectures [Vaswani *et al.*, 2017] and possess a large number of parameters, enabling them to generalize across a broad range of general tasks [Likhomanenko *et al.*, 2021; Touvron *et al.*, 2021; Dosovitskiy *et al.*, 2021; Min *et al.*, 2022]. However, achieving optimal performance on specific tasks typically necessitates fine-tuning these pre-trained models. Despite the formidable capabilities of LLMs, the fine-tuning process is resource-intensive, requiring significant computational power, storage, and time due to the large scale of model parameters involved. Fine-tuning all the parameters of a large language model, known as full fine-tuning, is highly computationally expensive. To reduce the computational cost, various parameter-efficient fine-tuning (PEFT) methods have been proposed [Ding *et al.*, 2023; Houlsby *et al.*, 2019; Lester *et al.*, 2021; Li and Liang, 2021; Hu *et al.*, 2022], which only fine-tune a small number of (extra) model parameters. A fundamental component of transformers is the attention mechanism, particularly the interactions among the query matrix $\mathbf{W}_q$ (Wq), the key matrix $\mathbf{W}_k$ (Wk), and the value matrix $\mathbf{W}_v$ (Wv).

During the fine-tuning of LLMs involving the attention mechanism, two interesting phenomena have been observed[1]: (1) *Unequal Importance of Attention Matrices*—optimizing the $\mathbf{W}_v$ is pivotal for enhancing performance, significantly more so than adjustments to the $\mathbf{W}_k$, which exhibit limited impact on the outcomes. Additionally, fine-tuning only the $\mathbf{W}_q$ and $\mathbf{W}_v$ often yields results that are comparable to or surpass those achieved by fine-tuning all three matrices $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$, which also reduces the number of tunable attention parameters by approximately 1/3, offering computational benefits (Section 3). (2) *Attention Matrices with Customized Learning Rate Lead to Better Convergence*—using the same learning rate for $\mathbf{W}_q\&\mathbf{W}_k$ and $\mathbf{W}_v$ is not optimal for efficient convergence. In fact, it is essential to apply distinct learning rates for the $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ components to ensure optimal fine-tuning performance. Specifically, the learning rate for $\mathbf{W}_v$ should generally be higher than that for $\mathbf{W}_q$ and $\mathbf{W}_k$ to facilitate efficient convergence (Section 4).

While certain empirical guidelines, such as the original

[*] Work done during the collaborative project with XiaoMi.

[†] Corresponding Author.

[1] Extended version and code, are available at https://github.com/XiaoMi/EfficientFT.

Low-Rank Adaptation (LoRA) [Hu *et al.*, 2022], explore which weight matrices in transformers are suitable for the application of LoRA, comprehensive theoretical analyses of these phenomena are still limited. This includes aspects such as selecting appropriate weight types for fine-tuning and optimizing learning rate settings. Reflecting on the attention equation itself (Section 2): (1) In linear algebra, two matrices multiplied without an intermediate activation can be equivalent to a single matrix. Some studies [Noci *et al.*, 2022; Bao *et al.*, 2024] often treat $\mathbf{W}_q$ and $\mathbf{W}_k$ as a single unit ($\mathbf{W}_{qk} = \mathbf{W}_q\mathbf{W}_k^T$), however, the benefits of fine-tuning $\mathbf{W}_q\&\mathbf{W}_v$ alone have yet to be further clarified. (2) Considering the scenario where the values of $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ approach zero, the gradients of $\mathbf{W}_q\&\mathbf{W}_k$ tend to diminish towards zero. In contrast, the gradient of $\mathbf{W}_v$ remains non-zero due to the influence of softmax normalization. Driven by the above motivations, this paper delves into the issue from the following two perspectives.

- **Generalization: advantages of fine-tuning $\mathbf{W}_q\&\mathbf{W}_v$ over $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$ together.** We perform a thorough theoretical analysis to demonstrate the advantages. To be more specific, we employ information-theoretic approaches [Xu and Raginsky, 2017; Polyanskiy and Wu, 2019; Wang and Mao, 2022; Zhu *et al.*, 2024] to establish the generalization bounds of fine-tuning pre-trained models with attention mechanism (See **Theorem 1** for details). This indicates that fine-tuning $\mathbf{W}_q\&\mathbf{W}_v$ instead of $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$ reduces the number of parameters, while improving generalization bounds and potentially providing memory benefits.
- **Optimization: convergence analysis of attention mechanism with varying learning rate settings.** To further investigate the aforementioned phenomena, we examine the optimization process of the attention mechanism. First, we discuss the learning dynamics in transformers in **Case 1**, suggesting that $\mathbf{W}_v$ may experience instances of inefficient learning during downstream task fine-tuning. This naturally leads to the hypothesis that accelerating the learning of $\mathbf{W}_v$ in the early stages could potentially induce $\mathbf{W}_k$ and $\mathbf{W}_q$ to begin learning earlier. Additionally, by using scaling arguments for large width-n networks [Yang *et al.*, 2022; Hayou *et al.*, 2024b], we illustrate (**Theorem 2**) that the feature learning of attention mechanism is efficient when the learning rate for $\mathbf{W}_v$ should be generally much larger than that of $\mathbf{W}_q\&\mathbf{W}_k$ in fine-tuning.

Building on our experimental and theoretical insights, one can develop new algorithms to improve the effectiveness (e.g., storage, and time) of fine-tuning. Experimental results for our strategy (in Section 5) on benchmark datasets [Wang *et al.*, 2018] and open source pre-trained models [Liu *et al.*, 2019; AI@Meta, 2024] verify that the method can visibly influence fine-tuning efficiency. We do not make direct comparisons with various parameter-efficient fine-tuning methods, as our strategy is primarily intended to demonstrate how theoretical analysis can effectively guide experimental procedures.

A summary of the main theoretical analyses. According to the traditional statistical learning viewpoint, performance can be defined by the sum of optimization error and generalization error. Our theoretical analyses in Sections 3 and 4 correspond to generalization and optimization, respectively. In Section 3 (generalization, storage-friendly), we give **Theorem 1** (information-theoretic generalization bounds), showing that with the same r value, fine-tuning $\mathbf{W}_q\&\mathbf{W}_v$ consistently achieves results comparable to or even surpassing those of fine-tuning $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$. This reduces the number of parameters for the same r, while improving generalization bounds and potentially providing memory benefits. In Section 4 (optimization, time-friendly), we discuss the learning dynamics in fine-tuning attention mechanism, and we illustrate (**Theorem 2**) that the feature learning of attention mechanism is efficient when the learning rate for $\mathbf{W}_v$ should be generally much larger than that of $\mathbf{W}_q\&\mathbf{W}_k$ in fine-tuning. Building on our experimental and theoretical insights, one can develop new algorithms to improve the effectiveness (e.g., storage, and time) of fine-tuning.

2 Preliminaries and Background

In this section, we first describe the core components of our study by reviewing some basic notations. The transformer model serves as the backbone of most state-of-the-art pretrained models. For clarity, we briefly outline its key equations, focusing on the self-attention function, as follows.

Self-attention. Given a sequence of m vectors $\mathbf{C} \in \mathbb{R}^{m \times d_{in}}$ over which we would like to perform attention and a query vector $\mathbf{x} \in \mathbb{R}^{d_{in}}$, that is, the input is $[\mathbf{C}, \mathbf{x}] \in \mathbb{R}^{(m+1)\times d_{in}}$. Conventional attention can be expressed as[2]:

$$\begin{aligned}&\mathrm{Attn}(\mathbf{x}\mathbf{W}_q, \mathbf{C}\mathbf{W}_k, \mathbf{C}\mathbf{W}_v)\\&= \mathrm{softmax}\left(\frac{\mathbf{x}\mathbf{W}_q\mathbf{W}_k^T\mathbf{C}^T}{\sqrt{d_{out}}}\right)\mathbf{C}\mathbf{W}_v,\end{aligned} \tag{1}$$

where $\mathbf{W}_q$,$\mathbf{W}_k$,$\mathbf{W}_v \in \mathbb{R}^{d_{in}\times d_{out}}$ are query, key and value (projection) matrices.

A unified framework for parameter-efficient fine-tuning. Building on [He *et al.*, 2022], we consider a unified framework that establishes connections among various PEFT methods. Specifically, we reinterpret these methods as modifications applied to specific hidden states within pre-trained models, the composition function can be written as:

$$\mathbf{h} \leftarrow l_1\mathbf{h} + l_2\Delta\mathbf{h}, \tag{2}$$

where l_1, l_2 are coefficients, $\mathbf{h}$ is denoted as the hidden representation to be directly modified and $\Delta\mathbf{h}$ is a modification vector. Moreover, $\mathbf{h}$ and $\mathbf{x}$ can represent the attention output and input respectively. Here, we present two special cases:

LoRA. LoRA [Hu *et al.*, 2022] injects trainable low-rank matrices into transformer layers to approximate the weight updates. Instead of directly adjusting the full weight matrix $\mathbf{W} \in \mathbb{R}^{d_{in}\times d_{out}}$, LoRA represents its update with a low-rank decomposition $\mathbf{W} + \Delta\mathbf{W} = \mathbf{W} + \mathbf{A}\mathbf{B}$, where $\mathbf{A} \in \mathbb{R}^{d_{in}\times r}, \mathbf{B} \in \mathbb{R}^{r\times d_{out}}$ are tunable parameters. For a specific input $\mathbf{x}$, LoRA modifies the projection output $\mathbf{h}$ as (where $s \geq 1$ is a tunable scalar hyperparameter):

$$\mathbf{h} \leftarrow \mathbf{h} + s\Delta\mathbf{h}, \quad \Delta\mathbf{h} := \mathbf{x}\mathbf{A}\mathbf{B}. \tag{3}$$

[2] For simplicity, we focus on the last vector of input in a single-head self-attention. Our analysis is readily generalizable to multi-head self-attention.

Prefix tuning. Prefix tuning [Li and Liang, 2021] prepends r tunable prefix vectors to the keys and values of the attention mechanism at every layer. Specifically, two sets of prefix vectors $\mathbf{P}_k, \mathbf{P}_v \in \mathbb{R}^{r \times d_{out}}$ are concatenated with the original key $\mathbf{CW}_k$ and value $\mathbf{CW}_v$, attention is then applied to the prefixed keys and values as[3]:

$$\mathbf{h} \leftarrow (1 - \alpha(\mathbf{x})\mathbf{h} + \alpha(\mathbf{x})\Delta\mathbf{h},$$
$$\Delta\mathbf{h} := \text{softmax}(\mathbf{x}\mathbf{W}_q\mathbf{P}_k^T)\mathbf{P}_v \triangleq \text{softmax}(\mathbf{x}\mathbf{A})\mathbf{B}, \qquad (4)$$

where $\alpha(\mathbf{x}) = \frac{\sum_i \exp(\mathbf{x}\mathbf{W}_q\mathbf{P}_k^T)_i}{\sum_i \exp(\mathbf{x}\mathbf{W}_q\mathbf{P}_k^T)_i + \sum_j \exp(\mathbf{x}\mathbf{W}_q\mathbf{W}_k^T\mathbf{C}^T)_j}$ is a scalar that represents the sum of normalized attention weights on the prefixes. We derive a detailed equivalent form of Prefix tuning to connect it with LoRA in Appendix A.1.

Remark 1. *By defining $\mathbf{A} = \mathbf{W}_q\mathbf{P}_k^T, \mathbf{B} = \mathbf{P}_v$ in Eq.(4), we can establish a connection with LoRA in Eq.(3). Notably, if we replace the softmax attention with linear attention here, the two are equivalent to some extent. Intuitively, in the attention mechanism, $\mathbf{A}$ $(\mathbf{W}_q\mathbf{P}_k^T)$ is responsible for generating attention scores, while $\mathbf{B}$ $(\mathbf{P}_v)$ utilizes these attention scores to produce the target content. Therefore, during fine-tuning, query, key, and value are likely to exhibit varying degrees of importance. This may also provide theoretical insights for recent works [Zhu* et al.*, 2024; Hayou* et al.*, 2024a], which empirically observed an asymmetry where the project-down matrix $\mathbf{A}$ is responsible for extracting features from the input, while the project-up matrix $\mathbf{B}$ utilizes these features to generate the desired output in LoRA.*

Θ Notation. We use standard asymptotic notation to describe behavior as the width n grows, following conventions in [Yang *et al.*, 2022; Hayou *et al.*, 2024b]. Given sequences $c_n \in \mathbb{R}$ and $d_n \in \mathbb{R}^+$, we write $c_n = O(d_n)$ and $c_n = \Omega(d_n)$ to mean $c_n < \kappa d_n$ or $c_n > \kappa d_n$, respectively, for some constant $\kappa > 0$. We denote $c_n = \Theta(d_n)$ when both $c_n = O(d_n)$ and $c_n = \Omega(d_n)$ hold, implying that c_n and d_n grow at comparable rates. For vector sequences $c_n = (c_n^i)_{1 \leq i \leq k} \in \mathbb{R}^k$ (for some $k > 0$), we write $c_n = O(d_n)$ when $c_n^i = O(d_n^i)$ for all $i \in [k]$, and analogous notation applies for other asymptotic bounds. Finally, when the sequence c_n is a vector of random variables, convergence is understood to refer to convergence in the second moment (i.e., L_2 norm).

3 Advantages and Generalization Analysis

In this section, we show our first interesting observation (*Unequal Importance of Attention Matrices*) in fine-tuning the attention mechanism and the storage benefits of fine-tuning only $\mathbf{W}_q$ and $\mathbf{W}_v$ (Section 3.1). Next, we give mutual information based generalization bounds of fine-tuning only $\mathbf{W}_q$ and $\mathbf{W}_v$ (Section 3.2), providing a better generalization error.

3.1 Empirical Advantages

To explore the *Unequal Importance of Attention Matrices*, we focus our study on **adapting only the attention weights** for downstream tasks, while freezing the other modules to ensure simplicity and parameter efficiency. Furthermore, we investigate the impact of adapting different types of attention weight matrices in a Transformer, as outlined below. We present our empirical results using LoRA to fine-tune a set of language models (Roberta-base [Liu *et al.*, 2019] and Llama3.1-8b [AI@Meta, 2024]) across various benchmarks [Wang *et al.*, 2018]. Further details on the experimental setup and additional empirical results are in Appendix B.1.

Table 1 provides a detailed comparison of the impact of fine-tuning different weight matrices ($\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$) across various rank values r and weight update strategies in LoRA fine-tuning on tasks like SST2, QNLI, QQP, and MNLI. We can see a clear trend where solely updating the $\mathbf{W}_v$ matrix outperforms just learning the $\mathbf{W}_q, \mathbf{W}_k$ matrix. Interestingly, the combination of fine-tuning both $\mathbf{W}_q$ and $\mathbf{W}_v$ often leads to performance that matches or even exceeds that achieved by fine-tuning all three matrices $\mathbf{W}_q, \mathbf{W}_k$, and $\mathbf{W}_v$. This pattern is consistently observed across various tasks and rank values, further emphasizing the importance of these two matrices over $\mathbf{W}_k$ during fine-tuning.

Computational benefits. Here, we show that the reduced amount of adapted parameters by (roughly) $1/3$ provides computational gains. The key benefit of parameter-efficient method is to save memory during training, storage and communication [Lialin *et al.*, 2023]. Fine-tuning $\mathbf{W}_q \& \mathbf{W}_v$ alone as opposed to both $\mathbf{W}_q \& \mathbf{W}_v$ and $\mathbf{W}_k$ reduces the number of parameters by $1/3$, when the dimensions of $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ are the same. Moreover, we discuss in Appendix B.2 about why fine-tune $\mathbf{W}_q \& \mathbf{W}_v$ instead of $\mathbf{W}_k \& \mathbf{W}_v$.

3.2 Information-Theoretic Generalization Bounds

In the previous part, we establish that the *Unequal Importance of Attention Matrices* among $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ during fine-tuning. Some studies [Noci *et al.*, 2022; Bao *et al.*, 2024] often treat $\mathbf{W}_q$ and $\mathbf{W}_k$ as a single unit ($\mathbf{W}_{qk} = \mathbf{W}_q\mathbf{W}_k^T$), however, the benefits of fine-tuning $\mathbf{W}_q \& \mathbf{W}_v$ alone, rather than fine-tuning $\mathbf{W}_q \& \mathbf{W}_v$, and $\mathbf{W}_k$ together, have yet to be further clarified. Therefore, we will further analyze this issue from an information-theoretic generalization perspective.

Recently, information-theoretic generalization bounds [Xu and Raginsky, 2017; Russo and Zou, 2019; Steinke and Zakynthinou, 2020; Wang and Mao, 2022] have been introduced to analyze the expected generalization error of learning algorithms. A key benefit of these bounds is that they depend not only on the data distribution but also on the specific algorithm, making them an ideal tool for studying the generalization behavior of models trained using particular algorithms.

Generalization error. We let $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ be the instance space and μ be an unknown distribution on $\mathcal{Z}$, specifying random variable Z. Here, $\mathcal{X}$ denotes the feature space and $\mathcal{Y}$ is the label space. Suppose one observes a training set $S_N \triangleq (Z_1, ..., Z_N) \in \mathcal{Z}^N$, with N i.i.d. training examples drawn from μ. In the information-theoretic analysis framework, we let $\mathcal{W}$ be the space of hypotheses related to the model, and a stochastic learning algorithm $\mathcal{A}$ which takes the training examples S_N as its input and outputs a hypothesis $W \in \mathcal{W}$ according to some conditional distribution $Q_{W|S_N}$. Given a loss function $\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}^+$, where $\ell(w, Z)$ measures the "unfitness" or "error" of any $Z \in \mathcal{Z}$ with respect

[3] Without loss of generalization, we ignore the softmax scaling factor for ease of notation.

	Weight Type	$\mathbf{W}_q$	$\mathbf{W}_k$	$\mathbf{W}_v$	$\mathbf{W}_q, \mathbf{W}_k$	$\mathbf{W}_q, \mathbf{W}_v$	$\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$
SST2(R)	$r = 4$	0.904	0.902	0.913	$\underline{\underline{0.919}}$	$\underline{\mathbf{0.920}}$	$\underline{\mathbf{0.920}}$
	$r = 8$	0.914	0.906	$\underline{\underline{0.918}}$	0.915	$\underline{0.919}$	**0.922**
	$r = 16$	0.907	0.905	0.916	$\underline{\underline{0.917}}$	$\underline{0.921}$	**0.923**
QNLI(R)	$r = 4$	0.854	0.835	$\underline{\underline{0.878}}$	0.866	**0.888**	$\underline{0.887}$
	$r = 8$	0.857	0.841	$\underline{\underline{0.875}}$	0.866	$\underline{0.889}$	**0.895**
	$r = 16$	0.854	0.840	$\underline{\underline{0.875}}$	0.867	$\underline{\mathbf{0.890}}$	$\underline{\mathbf{0.890}}$
QQP(R)	$r = 4$	0.812	0.804	$\underline{\underline{0.828}}$	0.823	$\underline{0.838}$	**0.843**
	$r = 8$	0.812	0.806	$\underline{\underline{0.828}}$	0.823	$\underline{0.840}$	**0.844**
	$r = 16$	0.812	0.804	$\underline{\underline{0.831}}$	0.823	$\underline{0.839}$	**0.844**
QQP(L)	$r = 8$	0.864	0.845	0.865	$\underline{\underline{0.866}}$	$\underline{\mathbf{0.874}}$	$\underline{\mathbf{0.874}}$
	$r = 16$	0.864	0.845	$\underline{\underline{0.869}}$	0.867	$\underline{\mathbf{0.874}}$	$\underline{\mathbf{0.874}}$
MNLI(R)	$r = 4$	0.748	0.733	$\underline{\underline{0.807}}$	0.772	$\underline{0.820}$	**0.828**
	$r = 8$	0.749	0.733	$\underline{\underline{0.809}}$	0.778	$\underline{0.820}$	**0.827**
	$r = 16$	0.750	0.734	$\underline{\underline{0.810}}$	0.780	$\underline{0.824}$	**0.828**
MNLI(L)	$r = 8$	0.802	0.660	$\underline{\underline{0.862}}$	0.814	$\underline{\mathbf{0.871}}$	$\underline{\mathbf{0.871}}$
	$r = 16$	0.803	0.663	$\underline{\underline{0.863}}$	0.815	$\underline{\mathbf{0.871}}$	$\underline{\mathbf{0.871}}$

Table 1: Performance comparison across different r values and weight types. To enable a fair comparison, we initialize the weights for all tasks with the original pretrained weights. Test accuracy of Roberta-base (R) and Llama3.1-8b (L) fine-tuning on SST2, QNLI, QQP, MNLI, with sequence length T = 128 and half precision (FP16). All values are averaged over 3 random seeds. The best result is shown in **bold**, the second best result is shown in $\underline{\text{underline}}$, and the third best result is shown with double $\underline{\underline{\text{underlines}}}$.

to a hypothesis $w \in \mathcal{W}$. We take ℓ as a continuous function and assume that ℓ is differentiable almost everywhere with respect to w. The goal of learning is to find a hypothesis w that minimizes the population risk, and for any $w \in \mathcal{W}$, the population risk is defined as $L_\mu(w) \triangleq \mathbb{E}_{Z \sim \mu}[\ell(w, Z)]$. However, since only can partially observe μ via the sample S_N, we instead turn to use the empirical risk, defined as $L_{S_N}(w) \triangleq \frac{1}{N}\sum_{i=1}^{N} \ell(w, Z_i)$. Then the expected generalization error of $\mathcal{A}$ is defined as

$$\widetilde{\text{error}}(\mathcal{A}) \triangleq \mathbb{E}_{W,S_N}[L_\mu(W) - L_{S_N}(W)],$$

where the expectation is taken over $(S_N, W) \sim \mu^N \otimes Q_{W|S_N}$.

Consider the following variations of fine-tuning algorithms: tuning both $\mathbf{W}_k$ and $\mathbf{W}_q\&\mathbf{W}_v$ matrices (as in classic attention mechanism in fine-tuning), tuning only $\mathbf{W}_q\&\mathbf{W}_v$:

Definition 1 (Fine-tuning algorithms)**.** *Recalling **A unified framework for parameter-efficient fine-tuning**, we can model the fine-tuning process of the attention mechanism as* $\mathbf{h} + \Delta\mathbf{h} = \mathbf{x}\mathbf{W} + \mathbf{x}\Delta\mathbf{W}$. *Let* $\mathbf{W} = \{\mathbf{W}_i\}_{i=1}^{L}$ *be a set of abstract parameter matrices related to a pretrained model, where each* $\mathbf{W}_i$ *is associated with the parameters* $\mathbf{W}_q^i, \mathbf{W}_k^i, \mathbf{W}_v^i$. *The indices* $1, ..., L$ *represent the layers of the model where these parameters are to be fine-tuned. Let* $\mathcal{I} \subseteq \{1, ..., L\}$ *denote the subset of layers selected for fine-tuning. Given a fine-tuning training set* S_N*, let* r *denote the chosen lora-rank, and assume each tuned parameter is quantized to* q *bits. Define the following algorithmic frameworks for selecting an adaptation* $\Delta\mathbf{W} = \{\Delta\mathbf{W}_i\}_{i=1}^{L}$ *(with other details left open to choice). (1)* $\mathcal{A}_{QKV}$*: For each* $i \in \mathcal{I}$*, optimize* $\{\mathbf{W}_q^i, \mathbf{W}_k^i, \mathbf{W}_v^i\}_{i \in \mathcal{I}}$ *to fit the data* S_N*. (2)* $\mathcal{A}_{QV}$*: For each* $i \in \mathcal{I}$*, optimize* $\{\mathbf{W}_q^i, \mathbf{W}_v^i\}_{i \in \mathcal{I}}$ *to fit the data* S_N*.*

Then we use the information-theoretic generalization framework to bound the generalization error:

Theorem 1 (Generalization bounds on adapting $\mathbf{W}_q\&\mathbf{W}_v$ and/or $\mathbf{W}_k$)**.** *Consider the algorithms of **Definition 1**. Assume the loss* $\ell(\mathbf{W}, Z)$ *is R-subGaussian under* $(\Delta\mathbf{W}, Z) \sim P_{\Delta\mathbf{W}|\mathbf{W}} \times \mu$*. Then (See Appendix A.2 for a proof),*

$$\widetilde{error}(\mathcal{A}_{QV}) \leq \sqrt{\frac{4R^2}{N} qr \sum_{i \in \mathcal{I}} (d_{in} + d_{out})},$$

$$\widetilde{error}(\mathcal{A}_{QKV}) \leq \sqrt{\frac{6R^2}{N} qr \sum_{i \in \mathcal{I}} (d_{in} + d_{out})},$$

where $\mathbf{W}_q^i, \mathbf{W}_k^i, \mathbf{W}_v^i \in \mathbb{R}^{d_{in} \times d_{out}}$*.*

Remark 2 (Discussion of the advantages)**.** *We can evaluate the empirical risk* (L_{S_N}) *by observing the model's performance on the dataset we have. If the generalization error (**Theorem 1**) is determined, it is at least possible to estimate the population risk* (L_μ)*. This generalization bound increases with the number of parameters being tuned, which grows as a function of* r *and the dimensions of the parameter matrices. In Table 1, we know that with the same* r *value, fine-tuning* $\mathbf{W}_q\&\mathbf{W}_v$ *consistently achieves results comparable to or even surpassing those of fine-tuning* $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$*. This reduces the number of parameters for the same* r*, while **improving generalization bounds and potentially providing memory benefits**.*

4 Convergence Analysis in Optimization

In Section 3, we have already demonstrated the generalization performance of the attention mechanism during fine-tuning. Our focus now shift toward optimizing convergence efficiency. Some optimization observations have also been reported in previous works [Hu *et al.*, 2022; Li *et al.*, 2023; He *et al.*, 2024], such as: [Li *et al.*, 2023] provide theoretical analyses of learning dynamics in transformers and observes a roughly two-stage process of self-attention. [He *et al.*, 2024] empirically show that the attention mechanism, particularly the value vector, stores the largest amount of memories and has the greatest influence during fine-tuning. However, there is not yet a satisfactory explanation for why this phenomenon occurs or how it can be effectively leveraged. In this section, we will explore these questions in more depth.

4.1 An Insight into Inefficient Learning

We first discuss the optimization process of attention mechanism in the following simple case.

Case 1. *Omitting the scale factor for qualitative analysis in Eq.(1), we obtain:*

$$Attn(\mathbf{x}\mathbf{W}_q, \mathbf{C}\mathbf{W}_k, \mathbf{C}\mathbf{W}_v) = softmax\left(\mathbf{x}\mathbf{W}_q\mathbf{W}_k^T\mathbf{C}^T\right)\mathbf{C}\mathbf{W}_v.$$

Intuitively, if $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$ are initialized as random matrices close to zero and trained simultaneously, then in the initial step, $\nabla_{\mathbf{W}_k}L(\nabla_{\mathbf{W}_q}L)$ contains the term $\mathbf{W}_q(\mathbf{W}_k)$, which is close to 0. By contrast, $\nabla_{\mathbf{W}_v}L$ contains the softmax-normalized attention weights. Therefore, during the initial steps (in training), $\mathbf{W}_v$ intuitively grows at a much faster rate than $\mathbf{W}_k(\mathbf{W}_q)$.

The work of [Li *et al.*, 2023] empirically exhibits **Case 1** with an approximately two-stage phenomenon: (1) In stage 1 (initial steps), the norms of $\mathbf{W}_k$ and $\mathbf{W}_q$ remain close to zero across all layers, while the norm of $\mathbf{W}_v$ increases significantly, accompanied by rapid changes in its orientation. (2) In stage 2, the norms of $\mathbf{W}_k$ and $\mathbf{W}_q$ begin to grow significantly, though much later than the $\mathbf{W}_v$ matrices. Briefly, in this case, $\mathbf{W}_v$ reaches a certain level of learning during training before $\mathbf{W}_k$ and $\mathbf{W}_q$ begin to learn. This suggests that when fine-tuning the model for downstream tasks, there may also be instances of inefficient learning in $\mathbf{W}_v$. Additionally, is there a fine-tuning strategy that could facilitate more effective learning for downstream tasks? **For instance, accelerating the learning of $\mathbf{W}_v$ in the early stages could potentially induce earlier learning in $\mathbf{W}_k$ and $\mathbf{W}_q$.**

Next, we present the second interesting phenomenon *Attention Matrices with Customized Learning Rate Lead to Better Convergence*. We use the General Language Understanding Evaluation (GLUE, [Wang *et al.*, 2018]) to evaluate the fine-tuning performance of different fine-tuning strategies, which consists of several language tasks that evaluate the understanding capabilities of language models. Using LoRA, we fine-tune Roberta-base from the RoBERTa family [Liu *et al.*, 2019] and Llama3.1-8b [AI@Meta, 2024] on MNLI, QQP, QNLI, and SST2 tasks with varying learning rates (η_{QK}, η_V) to identify the optimal combination. Other empirical details are provided in Appendix B.1 and we evaluate the LLaMA3.1-8B model on more complex benchmarks in Appendix B.3. We present our empirical results using LoRA to fine-tune language models, as visualized in the heatmaps (Figure 1 and Figure 2).

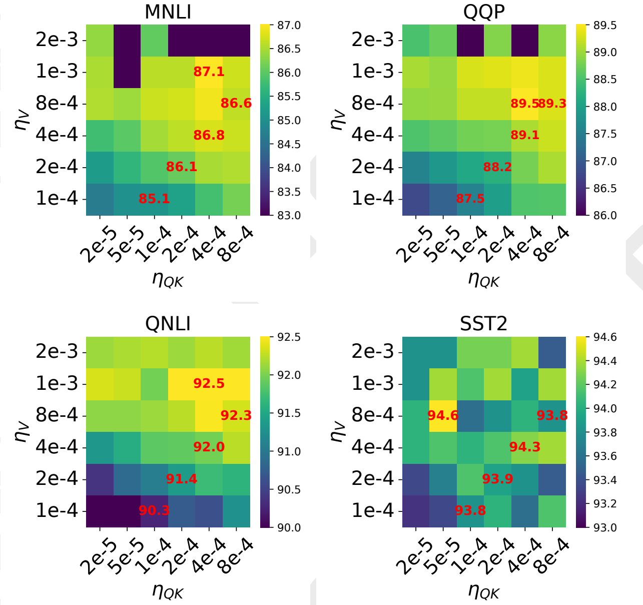

Figure 1: The test accuracy of RoBERTa-base fine-tuning was evaluated over 3 epochs for MNLI, QQP, and QNLI, and 6 epochs for SST-2, with a sequence length $T = 128$ and using half-precision (FP16). The LoRA hyperparameters were set to $\alpha = r = 8$. All reported values represent the average results across 3 random seeds. We use red color to highlight (1) the best overall accuracy and (2) the values where $\eta_V/\eta_{QK} = 1$. For better visualization, when accuracy is lower than a fixed threshold, we set it to threshold.

In Figure 1 and Figure 2 (Appendix B.3), we observe that (1) test accuracy consistently reaches its maximum for certain sets of learning rates where $\eta_{QK} < \eta_V$, outperforming the standard practice of setting η_{QK} and η_V equal. (2) More interestingly, the gap between the optimal choice of learning rates overall and the optimal choice when $\eta_{QK} = \eta_V$ varies across different tasks. This is probably due to the fact that harder task (like MNLI) requires more efficient feature learning. We compare two optimal learning rate (η_{QK}, η_V) settings in Figure 2 (Left), the $\eta_V >> \eta_{QK}$ setting has a better convergence than $\eta_V = \eta_{QK}$ setting in Figure 2 (Right).

It is also important to note that due to limited computational resources in our experiments, we use a sequence length of $T = 128$ and fine-tune for only 3 epochs on MNLI and QQP. Therefore, it is expected that our test accuracies may be lower than those reported by [Hu *et al.*, 2022], where the authors fine-tune RoBERTa-base with a sequence length of $T = 512$ (for MNLI) and for more epochs (30 for MNLI). We do not include confidence intervals for clearer visualization, however, the fluctuations remain within acceptable limits. See Figure 2 (Right) for instance. In Appendix B.3, we provide additional results including the training loss.

4.2 Convergence Analysis for Learning Rate

It naturally raises the question of why η_{QK} and η_V should be set differently. In practice, the large width (embedding dimension) of state-of-the-art models makes it valuable to examine training dynamics as width approaches infinity.

Starting with a Toy setting. Revisiting **Definition 1**, we have $\Delta \mathbf{h} = \mathrm{softmax}(\mathbf{xA})\mathbf{B}$. In the case of a linear attention mechanism, we instead have $\Delta \mathbf{h} = \mathbf{xAB}$. Then consider the following toy setting:

$$f(x) = x(W^* + a^T b),$$

where $W^* \in \mathbb{R}^{n\times 1}$ are the fixed[4] pre-trained weights, $b \in \mathbb{R}, a \in \mathbb{R}^{1\times n}$ are adaptation weights, $x \in \mathbb{R}^n$ is the model input (This corresponds to $r = 1$ in **Definition 1**). The training goal is to minimize the loss $\mathcal{L}(\theta) = \frac{1}{2}(f(x) - y)^2$ where $\theta = (a, b)$ and (x, y) is an input-output datapoint[5]. Similar to LoRA, we generally aim to initialize the product $a^T b$ to zero, ensuring that fine-tuning starts from the pre-trained model. This requires at least one of the weights, a (related to$\mathbf{W}_q \& \mathbf{W}_k$) or b (related to$\mathbf{W}_v$), to be initialized to zero. If both are initialized to zero, $\mathbf{W}_q \& \mathbf{W}_k$ learning cannot occur efficiently in init steps, as discussed in Section 4.1 (More detailed initialization settings are in Appendix A.3).

And we assume that $x = \Theta(1)$, meaning that the input coordinates remain of the same order as the width increases. In the subsequent analysis, we examine how the fine-tuning dynamics evolve as the model width n increases.

To streamline the analysis, we assume $W^* = 0$, a common simplification that can be applied without loss of generality. This assumption is implemented by setting $\hat{y} = y - xW^*$. We denote the fine-tuning step by using subscript t. Let $U_t = f_t(x) - y$, the gradients are then computed as:

$$\frac{\partial \mathcal{L}}{\partial a_t} = xU_t b_t, \quad \frac{\partial \mathcal{L}}{\partial b_t} = x a_t^T U_t.$$

And at step t with learning rate $\eta_a, \eta_b > 0$, we have

$$\Delta f_t \triangleq f_t(x) - f_{t-1}(x) = -\underbrace{\eta_a ||x||^2 U_{t-1} b_{t-1}^2}_{\delta_t^1}$$
$$-\underbrace{\eta_b (x a_{t-1}^T)^2 U_{t-1}}_{\delta_t^2} + \underbrace{\eta_a \eta_b ||x||^2 (x a_{t-1}^T) U_{t-1}^2 b_{t-1}}_{\delta_t^3}.$$

Remark 3. *The output update is influenced by three key terms. The first two items δ_t^1, δ_t^2 (order one in η_a/η_b) represent linear contributions to the update, meaning they result from changes in the model output when either a is updated with b held constant, or vice versa. The last item δ_t^3 (order two in $\eta_a \eta_b$) corresponds to a multiplicative update that captures the combined effects of changes in both a and b. As we scale the width*[6]*, **the desirable feature updates are such that** $\Delta f_t = \Theta(1)$, ensuring they remain unaffected by this scaling (the updates do not explode with width, see x for more details). Ideally, we aim for both δ_t^1 and δ_t^2 to be $\Theta(1)$. If this condition isn't met, it indicates that either a or b is not being updated efficiently. For example, if $\delta_t^1 = o(1)$, it suggests that as $n \to \infty$, the model behaves as if a is essentially fixed, with only b being trained. We say that the **feature learning in the attention mechanism is efficient** when $\delta_t^i = \Theta(1)$ for $i \in \{1, 2\}$ and all $t > 1$, it means that both a and b parameter updates significantly contribute to the change in $f_t(x)$. We will see that when δ_t^1, δ_t^2 are $\Theta(1)$, the term δ_t^3 is also $\Theta(1)$.*

Let us assume that we train the model with gradient descent with learning rate $\eta_a = \Theta(n^{c_a}), \eta_b = \Theta(n^{c_b})$ for some $c_a, c_b \in \mathbb{R}$. In the study by [Yang *et al.*, 2022], it is noted that the training dynamics primarily involve operations such as matrix-vector products and the summation of vectors or scalars. Given the nature of these operations, it is easy to see that any quantity in the training dynamics should be of order n^γ for some $\gamma \in \mathbb{R}$. We write $v = \Theta(n^{\gamma[v]})$, for any quantity v in the training dynamics. When v is a vector, we use the same notation when all entries of v are $\Theta(n^{\gamma[v]})$ (See Appendix A.4 for the formal definition of γ).

With reference to the method of [Hayou *et al.*, 2024b], we start from the initialization in **Starting with a Toy setting**, we have $f_0(x) = 0$. Feature learning of attention mechanism is efficient when $\delta_t^i = \Theta(1)$ for $i \in \{1, 2\}$ and all $t > 1$, and $f_t(x) = \Theta(1)$ for $t > 1$. This can be interpreted as:

$$\begin{cases} c_a + 1 + 2\gamma[b_{t-1}] = 0 \quad \left(\delta_t^1 = \Theta(1)\right) \\ c_b + 2\gamma[x a_{t-1}^\top] = 0 \quad \left(\delta_t^2 = \Theta(1)\right) \\ \gamma[x a_{t-1}^\top] + \gamma[b_{t-1}] = 0 \quad \left(f_{t-1}(x) = \Theta(1)\right), \end{cases}$$

which, after simple calculations, implies that $c_a + c_b = -1$. Notice that the above also leads to the $c_a + c_b + 1 + \gamma[x a_{t-1}^\top] + \gamma[b_{t-1}] = 0$ $(\delta_t^3 = \Theta(1))$. This is only a necessary condition. In the following section, we will provide theoretical conclusions in the toy model setting that offer guidance for real-world experiments.

Theorem 2 (Efficient fine-tuning in attention mechanism (Informal))**.** *In the case of **Starting with a Toy setting**, with $\eta_a = \Theta(n^{-1})$ and $\eta_b = \Theta(1)$, we have for all $t > 1$, $i \in \{1, 2, 3\}, \delta_t^i = \Theta(1)$. In other words, the feature learning of attention mechanism is efficient when $\eta_{QK}(\eta_a) = \Theta(n^{-1}), \eta_V(\eta_b) = \Theta(1)$. We denote η_V/η_{QK} as λ. We refer the reader to Appendix A.5 for more details on the proof.*

Remark 4. *In practice, **Theorem 2** implies that the learning rate for $\mathbf{W}_v$ should be generally much larger than that of $\mathbf{W}_q \& \mathbf{W}_k$ in fine-tuning. We verify that this scaling is valid for general neural network models in Section 4.1. Naturally, the optimal ratio λ depends on the architecture and the fine-tuning task through the constants in 'Θ'. This represents a limitation of the asymptotic results, as they do not provide insights into how the task and neural architecture influence these constants. We will further address this issue in future.*

5 An Example of Improving Fine-tuning

Based on all our exciting insights, it becomes intuitive to design lightweight attention-based fine-tuning improvements,

[4] Here, we primarily focus on the case of $\Delta\mathbf{W}$ to provide insightful theoretical results.

[5] To simplify the analysis, we assume that the fine-tuning dataset consists of a single sample, though our analysis can be easily generalized to multiple samples. All conclusions remain essentially valid when (a, b) are matrices.

[6] This property is generally satisfied in practice when the model width is large (e.g., $n \approx 800$ for Roberta-base and $n \approx 4000$ for Llama3.1-8b).

Method	**Trainable #Param (M)**	**RTE**	**STS-B**	**MRPC**	**CoLA**	**MNLI**	**SST-2**	**QQP**	**QNLI**
Before Fine-tune	0	45.12	-3.18	66.66	1.09	32.95	49.31	44.72	50.81
Full Fine-tune (QKV)	21.85	73.64	90.49	84.55	60.34	86.68	93.23	90.48	92.37
LoRA (QKV) $r = 8$	1.62	70.76	90.25	85.04	58.03	86.70	93.92	89.15	92.17
LoRA (QKV) $r = 16$	2.07	70.39	90.25	86.03	58.04	86.78	93.92	89.26	92.18
DoRA (QKV) $r = 8$	1.06	70.75	90.39	85.78	56.79	86.73	93.58	89.34	92.22
DoRA (QKV) $r = 16$	1.51	70.40	90.31	86.03	57.81	86.77	93.92	89.30	92.48
Full Fine-tune (QV) $\lambda = 2$	14.76	73.53	91.01	86.02	60.57	62.03	93.11	**90.56**	91.96
Full Fine-tune (QV) $\lambda = 4$	14.76	72.29	90.56	87.01	**61.88**	35.44	91.05	89.81	88.85
Full Fine-tune (QV) $\lambda = 8$	14.76	72.29	90.02	88.97	61.86	35.44	84.75	85.93	50.54
LoRA (QV) $r = 8, \lambda = 2$	1.48	71.84	90.37	86.02	58.54	86.85	94.03	89.47	92.33
LoRA (QV) $r = 8, \lambda = 4$	1.48	75.09	90.83	87.01	59.56	86.95	94.04	90.09	92.86
LoRA (QV) $r = 8, \lambda = 8$	1.48	76.13	90.75	88.97	**61.88**	86.93	93.46	90.01	92.34
LoRA (QV) $r = 16, \lambda = 2$	1.77	70.39	90.46	86.03	58.55	86.83	**94.38**	89.77	92.33
LoRA (QV) $r = 16, \lambda = 4$	1.77	76.17	**91.05**	87.99	60.06	87.19	94.03	90.30	92.73
LoRA (QV) $r = 16, \lambda = 8$	1.77	72.92	90.96	**89.95**	59.31	**87.31**	93.92	90.43	92.95
DoRA (QV) $r = 8, \lambda = 2$	0.90	71.12	90.29	87.01	58.54	87.08	93.96	89.60	92.60
DoRA (QV) $r = 8, \lambda = 4$	0.90	75.45	90.82	86.76	60.32	86.98	93.81	90.33	92.97
DoRA (QV) $r = 8, \lambda = 8$	0.90	70.76	90.38	87.75	57.01	87.12	94.15	90.45	92.48
DoRA (QV) $r = 16, \lambda = 2$	1.20	69.68	90.53	87.75	59.31	87.09	93.92	89.68	92.70
DoRA (QV) $r = 16, \lambda = 4$	1.20	76.16	90.77	88.48	60.84	86.96	94.15	90.34	**93.01**
DoRA (QV) $r = 16, \lambda = 8$	1.20	**77.26**	90.83	88.96	60.32	87.10	94.17	90.46	92.80

Table 2: Comparison of fine-tuning methods across GLUE benchmark. We report results on development set, Pearson correlation for STS-B, Matthew's correlation for CoLA, average accuracy for MNLI (matched and mismatched), and accuracy for other tasks. The best results on each dataset are shown in **bold** and the second best results are shown in underline. The QKV(QV) setting refers to fine-tuning $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v(\mathbf{W}_q, \mathbf{W}_v)$. It is noted that the total number of parameters in the Roberta-base model is 124.65M. λ means $\eta_V = \lambda\eta_Q$ and r is the LoRA rank, and a larger λ does not necessarily lead to better performance.

particularly for downstream tasks. To illustrate how theoretical analysis effectively guides experimental procedures, we propose an example method where we freeze the $\mathbf{W}_k$ and fine-tuning the $\mathbf{W}_q$&$\mathbf{W}_v$ using different learning rates. This procedure is reported in Figure 5. In Appendix B.2, we discuss **how to set the ratio λ?**

Experimental setup. We conduct experiments on widely adopted benchmark datasets [Wang *et al.*, 2018] and Roberta-base model [Liu *et al.*, 2019]. We selected mainstream baselines: Full Fine-tuning, LoRA [Hu *et al.*, 2022] and DoRA [Liu *et al.*, 2024]. Additionally, we adapt **only the attention weights** for downstream tasks, keeping the other modules frozen to maintain simplicity and validate the theoretical guidance through experiments. In our experiments, we evaluated the performance for λ values of 2, 4, and 8 (one can also determine a general optimal ratio through experiments, and even apply different settings across different layers of the model). We report the average results based on 3 random seeds, as shown in Table 2. The hyperparameter settings for the experiments can be found in Appendix B.1 and the base model performance for each task can be seen in Table 2 and Appendix B.3. We also extend ablation experiments on Mistral-7B [AI@Mistral, 2023] in Appendix B.3.

Results. We leverage our theoretical results (**Theorem 1** and **Theorem 2**) to enhance the efficiency of existing fine-tuning methods, such as Full Fine-tune, LoRA [Hu *et al.*, 2022] and DoRA [Liu *et al.*, 2024], on downstream tasks. As shown in Table 2, the improved fine-tuning approach not only outperforms the original version but also significantly reduces the number of parameters. For instance, on the MRPC task, *LoRA (QV) $r = 16, \lambda = 8$ (1.77M)* achieves better performance compared to *Full Fine-tune (QKV) (21.85M)* and *LoRA (QKV) $r = 16$ (2.07M)*. This series of experiments clearly demonstrates that our theoretical insights effectively enhance fine-tuning algorithms, particularly in terms of memory usage and optimization efficiency. Moreover, these theoretical results can guide the improvement of other fine-tuning algorithms and even aid in the design of more efficient ones.

6 Conclusion and Limitation

In this paper, we present our key findings in fine-tuning attention mechanism: *Unequal Importance of Attention Matrices* and *Attention Matrices with Customized Learning Rate Lead to Better Convergence*. While theoretical analysis of these phenomena is limited, this paper provides insights from two angles: *Generalization*—fine-tuning only $\mathbf{W}_q$ and $\mathbf{W}_v$ improves generalization and memory efficiency, and *Optimization*—using different learning rates enhances the efficiency of feature learning in the attention mechanism, leading to more effective fine-tuning. Our analysis provides a theoretical foundation for the configuration and improvement of lightweight algorithms in LLMs fine-tuning. However, further studies are required on how task type and architecture affect the optimal learning rate ratio λ. These studies will further deepen our understanding of attention-based fine-tuning.

Ethical Statement

There are no ethical issues.

Acknowledgments

We sincerely appreciate the anonymous reviewers for their valuable suggestions. This research was supported by National Natural Science Foundation of China (No.62476277), National Key Research and Development Program of China (NO. 2024YFE020320), CCF-ALIMAMA TECH Kangaroo Fund(No.CCF-ALIMAMA OF 2024008), and Huawei-Renmin University joint program on Information Retrieval. We also acknowledge the support provided by the fund for building worldclass universities (disciplines) of Renmin University of China and by the funds from Beijing Key Laboratory of Big Data Management and Analysis Methods, Gaoling School of Artificial Intelligence, Renmin University of China, from Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education, from Intelligent Social Governance Interdisciplinary Platform, Major Innovation & Planning Interdisciplinary Platform for the "DoubleFirst Class" Initiative, Renmin University of China, from Public Policy and Decision-making Research Lab of Renmin University of China, and from Public Computing Cloud, Renmin University of China.

References

[AI@Meta, 2024] AI@Meta. Llama 3.1 model card, 2024. Model Release Date: July 23, 2024.

[AI@Mistral, 2023] AI@Mistral. Mistral 7b model, 2023. Model Release Date: September 27, 2023.

[Bao *et al.*, 2024] Han Bao, Ryuichiro Hataya, and Ryo Karakida. Self-attention networks localize when qk-eigenspectrum concentrates. In *ICML*, 2024.

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

[Dosovitskiy *et al.*, 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[Hayou *et al.*, 2024a] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. The impact of initialization on lora finetuning dynamics. *arXiv preprint arXiv:2406.08447*, 2024.

[Hayou *et al.*, 2024b] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. In *ICML*, 2024.

[He *et al.*, 2022] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *ICLR*, 2022.

[He *et al.*, 2024] Haoze He, Juncheng Billy Li, Xuan Jiang, and Heather Miller. Sparse matrix in large language model fine-tuning. *arXiv preprint arXiv:2405.15525*, 2024.

[Houlsby *et al.*, 2019] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, pages 2790–2799, 2019.

[Hu *et al.*, 2022] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022.

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

[Li and Liang, 2021] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*, 2021.

[Li *et al.*, 2023] Yuchen Li, Yuanzhi Li, and Andrej Risteski. How do transformers learn topic structure: Towards a mechanistic understanding. In *ICML*, pages 19689–19729, 2023.

[Lialin *et al.*, 2023] Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. Scaling down to scale up: A guide to parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.15647*, 2023.

[Likhomanenko *et al.*, 2021] Tatiana Likhomanenko, Qiantong Xu, Jacob Kahn, Gabriel Synnaeve, and Ronan Collobert. slimipl: Language-model-free iterative pseudo-labeling. In *Proc. Interspeech 2021*, pages 741–745, 2021.

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

[Liu *et al.*, 2024] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *ICML*, 2024.

[Min *et al.*, 2022] Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Noisy channel language model prompting for few-shot text classification. In *ACL*, pages 5316–5330, 2022.

[Noci *et al.*, 2022] Lorenzo Noci, Stefanos Anagnostidis, Luigi Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurelien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. In *NeurIPS*, pages 27198–27211, 2022.

[Polyanskiy and Wu, 2019] Yury Polyanskiy and Yihong Wu. Lecture notes on information theory. Lecture Notes for 6.441 (MIT), ECE 563 (UIUC), STAT 364 (Yale), 2019.

[Russo and Zou, 2019] Daniel Russo and James Zou. How much does your data exploration overfit? controlling bias via information usage. *IEEE Transactions on Information Theory*, 66(1):302–323, 2019.

[Steinke and Zakynthinou, 2020] Thomas Steinke and Lydia Zakynthinou. Reasoning about generalization via conditional mutual information. In *Conference on Learning Theory*, 2020.

[Touvron *et al.*, 2021] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357, 2021.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[Wang and Mao, 2022] Ziqiao Wang and Yongyi Mao. Two facets of sde under an information-theoretic lens: Generalization of sgd via training trajectories and via terminal states. *arXiv preprint arXiv:2211.10691*, 2022.

[Wang *et al.*, 2018] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018.

[Xu and Raginsky, 2017] Aolin Xu and Maxim Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. In *NeurIPS*, pages 2524–2533, 2017.

[Yang *et al.*, 2022] Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. In *NeurIPS*, 2022.

[Zhu *et al.*, 2024] Jiacheng Zhu, Kristjan Greenewald, Kimia Nadjahi, Haitz Sáez de Ocáriz Borde, Rickard Brüel Gabrielsson, Leshem Choshen, Marzyeh Ghassemi, Mikhail Yurochkin, and Justin Solomon. Asymmetry in low-rank adapters of foundation models. *arXiv preprint arXiv:2402.16842*, 2024.