BEFT: Bias-Efficient Fine-Tuning of Language Models in Low-Data Regimes

Baichuan Huang[1], Ananth Balashankar[2]*, Amir Aminifar[1],
[1]Lund University, Sweden, [2]Google DeepMind, USA,
{baichuan.huang, amir.aminifar}@eit.lth.se, ananthbshankar@google.com

Abstract

Fine-tuning the bias terms of large language models (LLMs) has the potential to achieve unprecedented parameter efficiency while maintaining competitive performance, particularly in low-data regimes. However, the link between fine-tuning different bias terms (i.e., $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ in the query, key, or value projections) and downstream performance remains largely unclear to date. In this paper, we investigate the link between fine-tuning $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ with the performance of the downstream task. Our key finding is that *directly fine-tuning $\boldsymbol{b}_v$ generally leads to higher downstream performance in low-data regimes, in comparison to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$*. We extensively evaluate this unique property across a wide range of LLMs spanning encoder-only and decoder-only architectures up to 6.7B parameters (including bias-free LLMs). Our results provide strong evidence for the effectiveness of directly fine-tuning $\boldsymbol{b}_v$ across various downstream tasks. The implementation code is available at https://github.com/whubaichuan/BEFT.

1 Introduction

Fine-tuning pre-trained large language models (LLMs) for downstream tasks has gained a lot of attention over the past few years. Parameter-efficient fine-tuning (PEFT) methods have been widely studied (Ding et al., 2023; Yang et al., 2024) to reduce the fine-tuning overheads, such as computation cost, graphics processing unit (GPU) memory usage, and energy consumption. Among these PEFT techniques, bias-only fine-tuning—which involves updating only the bias terms of the LLMs (Zaken et al., 2022)—provides the potential for unprecedented parameter efficiency.

Bias-only fine-tuning offers out-of-the-box usability, without extra requirements for additional configuration and auxiliary reparameterization.

*contributed in an advisory role.

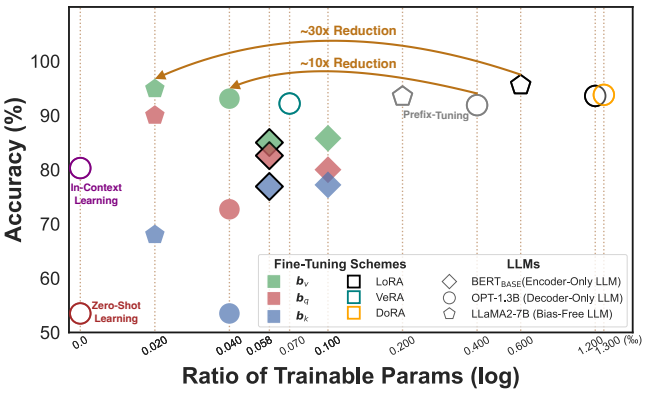


Figure 1: Fine-tuning $\boldsymbol{b}_v$ (green) leads to higher downstream performance, compared to $\boldsymbol{b}_q$ (red) and $\boldsymbol{b}_k$ (blue) (even combined with PEFT such as LoRA), on the SST-2 dataset with low-data regime (1000 training samples).

This is in contrast to the majority of the previous work in this domain, e.g., prefix tuning (Li and Liang, 2021), adapter tuning (Houlsby et al., 2019), LoRA (Hu et al., 2022), and GaLore (Zhao et al., 2024). Moreover, in low-data regimes, fine-tuning all bias terms has been shown to be competitive with full-parameter fine-tuning, despite updating a significantly smaller subset of full parameters (Zaken et al., 2022; Logan IV et al., 2022). Despite the advantages of bias-only fine-tuning, the relationship between fine-tuning different bias terms and downstream performance remains largely unclear (Ding et al., 2023; Weng et al., 2024).

In this paper, we investigate the link between fine-tuning different bias terms (i.e., $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ in the query, key, or value projections) and downstream performance. We observe that fine-tuning $\boldsymbol{b}_v$ generally leads to higher downstream performance in low-data regimes, compared to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$. For instance, for BERT$_{\text{BASE}}$ in Fig. 1, fine-tuning the value bias (green rhombus) leads to higher downstream performance than $\boldsymbol{b}_q$ (red rhombus) and $\boldsymbol{b}_k$ (blue rhombus). We then analyze the expressiveness of bias terms $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, $\boldsymbol{b}_v$, and provide further evidence supporting our observation.

We empirically validate the effectiveness of fine-tuning $\boldsymbol{b}_v$ versus $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$ on $\text{BERT}_{\text{BASE}}$ using the GLUE/SuperGLUE benchmarks (Wang et al., 2019b,a), without any post-hoc evaluation. We further extend our validation to autoregressive LLMs (such as OPT-1.3B) and bias-free LLMs (such as LLaMA2-7B) across various tasks. At the same time, our work is compatible and can be readily combined with PEFT methods, e.g., low-rank adaptation (LoRA) (Hu et al., 2022), vector-based random matrix adaptation (VeRA) (Kopiczko et al., 2024), weight-decomposed low-rank adaptation (DoRA) (Liu et al., 2024), and PiSSA (Meng et al., 2024). In particular, we show that directly fine-tuning $\boldsymbol{b}_v$ using LoRA leads to higher downstream performance compared to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$ in Fig. 1, with a substantially lower number of parameters. Our main contributions are:

- We investigate the link between fine-tuning $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ with the performance of the downstream task, both analytically and empirically. We study and shed light on the expressive power of bias terms $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ in the query, key, or value projections. Our key finding is that *directly fine-tuning $\boldsymbol{b}_v$ generally leads to higher downstream performance in low-data regimes, in comparison to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$, without requiring any post-hoc evaluation.*

- We extensively validate the effectiveness of directly fine-tuning $\boldsymbol{b}_v$ in low-data regimes, without relying on any post-hoc evaluation, across a wide set of LLMs, covering both encoder-only and decoder-only architectures with up to 6.7B parameters, including bias-free LLMs. Our experiments span a variety of downstream tasks, such as classification, multiple-choice, and generation tasks. Moreover, our results show that fine-tuning $\boldsymbol{b}_v$ using LoRA/VeRA/DoRA leads to higher downstream performance compared to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$.

2 Bias-Efficient Fine-Tuning (BEFT) for Scaled Dot-Product Attention

Given an LLM model with pre-trained parameters including weights and bias terms, the number of bias terms is substantially smaller than that of weights. In this section, we investigate the link between the bias terms $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ and the downstream performance. Following the procedure in (Zaken et al., 2022), we first fine-tune all the bias terms and then evaluate the change of different bias terms before and after fine-tuning, i.e., $\Delta\boldsymbol{b}_{\mathcal{T}}$, where $\boldsymbol{b}_{\mathcal{T}}$ captures the collection of the single type of bias across Transformer layers l from 1 to L, i.e., $\boldsymbol{b}_{\mathcal{T}} = \left\{\boldsymbol{b}_{\mathcal{T}}^{(l)}\right\}_{l=1}^{L}$. In bias-efficient fine-tuning (BEFT), we aim to identify the bias term, among query $\boldsymbol{b}_q$, key $\boldsymbol{b}_k$, and value $\boldsymbol{b}_v$ for scaled dot-product attention (SDPA), that has the largest change, denoted as the target bias term, for effective fine-tuning (rather than all bias terms). In other words, the target bias type $\mathcal{T}$ with the largest $\Delta(\boldsymbol{b}_{\mathcal{T}})$ is:

$$\mathcal{T} = \underset{\mathcal{T}\in\{q,\,k,\,v\}}{\arg\max}\,\{\Delta(\boldsymbol{b}_{\mathcal{T}})\}.$$

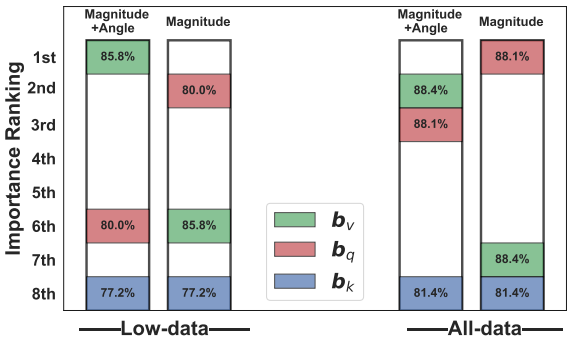
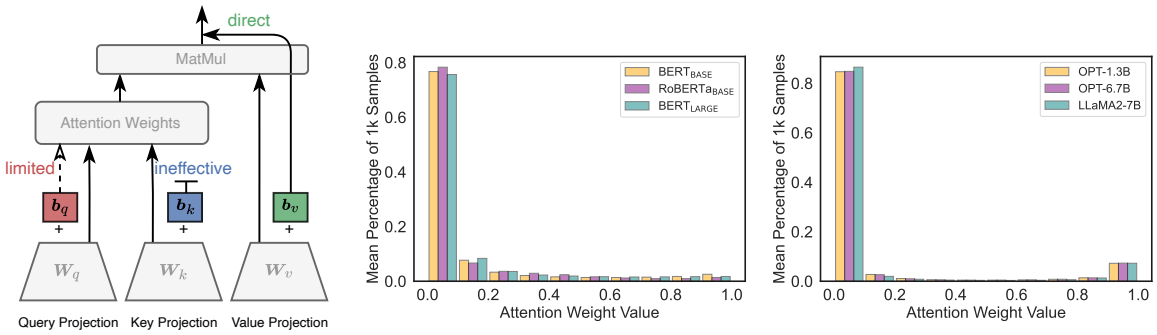


Figure 2: Importance ranking and accuracy (%) of fine-tuning different bias terms on SST-2 with low-data regime on $\text{BERT}_{\text{BASE}}$. We expect higher-ranked bias terms to achieve higher accuracy. Incorporating angular changes into magnitude changes effectively links the target bias term with the downstream performance.

In our empirical studies, we observe that the accuracy of fine-tuning $\boldsymbol{b}_v$ is higher than that of fine-tuning $\boldsymbol{b}_q$, especially in the low-data regime. In contrast, previous work (Zaken et al., 2022) considers only the magnitude of bias change and overall views the change in $\boldsymbol{b}_q$ is more than $\boldsymbol{b}_v$, which is inconsistent with our observation considering the downstream performance. To further investigate this and inspired by weight-parameter decomposition (Liu et al., 2024; Bini et al., 2025), we augment the magnitude changes with angular changes to study the link between fine-tuning different bias terms and downstream performance.

We denote the bias term at layer l before fine-tuning as $\boldsymbol{b}_{\mathcal{T}}^{(l),pre}$ and after fine-tuning as $\boldsymbol{b}_{\mathcal{T}}^{(l),post}$. We incorporate angular changes into magnitude changes by projection and then scale normalization. Overall, the change of the bias term across all layers is defined as follows (the derivation is in Appendix A.3):



(a) Expressiveness of $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$. (b) Distribution for Encoder-Only LLMs. (c) Distribution for Decoder-Only LLMs.

Figure 3: (a) Fine-tuning the investigated positions—$\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$—affects the expressiveness of the low-rank mapping differently: $\boldsymbol{b}_k$ is ineffective; $\boldsymbol{b}_q$ provides only limited improvement; $\boldsymbol{b}_v$ acts effectively as a direct addition after the SDPA output. The distributions of attention weights for (b) encoder-only and (c) decoder-only LLMs with the mean percentage of 1000 samples, showing inherent sharpness and high sparsity of attention weights.

$$\Delta(\boldsymbol{b}_{\mathcal{T}}) = \frac{1}{L}\sum_{l=1}^{L}\left(1 - \frac{\boldsymbol{b}_{\mathcal{T}}^{(l),pre}\cdot\boldsymbol{b}_{\mathcal{T}}^{(l),post}}{\max\left(\|\boldsymbol{b}_{\mathcal{T}}^{(l),pre}\|_2^2, \|\boldsymbol{b}_{\mathcal{T}}^{(l),post}\|_2^2\right)}\right).$$

As shown in Fig. 2, augmenting the magnitude with angle effectively links the target bias term with the downstream performance that can be achieved by fine-tuning $\boldsymbol{b}_v$, compared to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$. Similar patterns have been observed in our empirical evaluation presented in Section 3.2.

Next, we discuss the potential expressive power of the bias terms query $\boldsymbol{b}_q$, key $\boldsymbol{b}_k$, and value $\boldsymbol{b}_v$ for SDPA. Concurrent with our work, (Qiu et al., 2025) has attracted considerable attention from the research community by showing that gating after the value projection is more effective than after the query or key projection. This study further suggests that gating enhances the expressiveness of low-rank mappings in the attention module. Similarly, $\boldsymbol{b}_{\mathcal{T}}$ in BEFT can be interpreted as a linear variant of the additive gating introduced in (Qiu et al., 2025). As such, $\boldsymbol{b}_{\mathcal{T}}$ can potentially improve the expressiveness of low-rank mapping.

As shown in Fig. 3a, the query/key/value linear projections are based on $\boldsymbol{W_{q/k/v}}$ and $\boldsymbol{b_{q/k/v}}$, as well as the input $\boldsymbol{X}$. For bias-free LLMs that tend not to include bias terms in the attention module (Touvron et al., 2023), we can manually add $\boldsymbol{b_{q/k/v}}$ in their attention module, denoted as follows: $\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_q + \boldsymbol{b}_q, \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_k + \boldsymbol{b}_k, \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_v + \boldsymbol{b}_v$. Then, SDPA is calculated as follows:

$$\text{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}})\boldsymbol{V},$$

where d_k is the size of the key vector within one attention head; $\text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}})$ is the attention weights $\boldsymbol{A}$. Given $\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V} \in \mathbb{R}^{n\times d_k}$ (n is the sequence length) and softmax as row-wise, we exploit $\boldsymbol{q} \in \mathbb{R}^{1\times d_k}$ (one row of $\boldsymbol{Q}$). For $\boldsymbol{b}_k \in \mathbb{R}^{1\times d_k}$,

$$\boldsymbol{a} = \text{softmax}(\frac{\boldsymbol{q}\boldsymbol{K}^T}{\sqrt{d_k}}) = \text{softmax}(\frac{\boldsymbol{q}(\boldsymbol{K}+\boldsymbol{b}_k)^T}{\sqrt{d_k}}),$$

where $\boldsymbol{a} \in \mathbb{R}^{1\times n}$ is one row of attention weights $\boldsymbol{A}$ and $\boldsymbol{b}_k$ is broadcast across the rows of $\boldsymbol{K}$, so that each row of $\boldsymbol{K}$ is incremented by the same elements, while within each row the elements can differ. The constant $\boldsymbol{q}\boldsymbol{b}_{\boldsymbol{k}}{}^T \in \mathbb{R}^1$ is broadcast across the column of $\boldsymbol{q}\boldsymbol{K}^T$. Because softmax is shift-invariant, i.e., $\text{softmax}(\boldsymbol{q}\boldsymbol{K}^T) = \text{softmax}(\boldsymbol{q}\boldsymbol{K}^T + \boldsymbol{q}\boldsymbol{b}_{\boldsymbol{k}}{}^T)$, hence $\boldsymbol{b}_k$ will not improve expressiveness.

For $\boldsymbol{b}_q \in \mathbb{R}^{1\times d_k}$, $\boldsymbol{b}_q$ instead affects the attention weights after softmax. The Jacobian of softmax is:

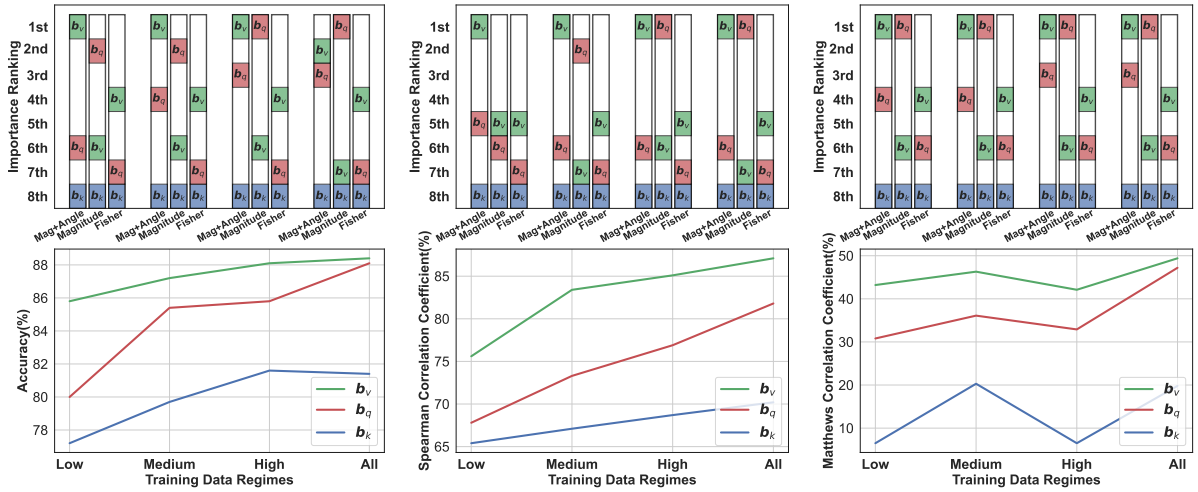
$$J = \text{diag}(\boldsymbol{a}) - \boldsymbol{a}^T\boldsymbol{a}, \quad |J_{i,j}| \longrightarrow 0, \forall i,j$$

where $J \in \mathbb{R}^{n\times n}$ and diag is the diagonal matrix. The diagonal element is $\boldsymbol{a}_i(1-\boldsymbol{a}_i)$ and the off-diagonal element is $-\boldsymbol{a}_i\boldsymbol{a}_j$. Moreover, because of $\boldsymbol{a}_i \in (0,1)$ and $\sum \boldsymbol{a}_i = 1$, the majority of $|J_{i,j}|$ approach zero especially in high-dimension d_k, due to the observed inherent sharpness and high sparsity of $\boldsymbol{A}$ in Fig. 3b and 3c (as well as Appendix-B.1-Fig. 7 and 8). Therefore, softmax is insensitive to the change of $\boldsymbol{b}_q$ and the capacity of $\boldsymbol{b}_q$ in improving the expressiveness is limited.

For $\boldsymbol{b}_v \in \mathbb{R}^{1\times d_k}$, because of $\sum \boldsymbol{a}_i = 1$, we have:

$$\boldsymbol{a}(\boldsymbol{V}+\boldsymbol{b}_v) = \boldsymbol{a}\boldsymbol{V} + \boldsymbol{a}(\boldsymbol{1}_n\boldsymbol{b}_v) = \boldsymbol{a}\boldsymbol{V} + \boldsymbol{b}_v, \quad (1)$$

where $\boldsymbol{b}_v$ is broadcast across the rows of $\boldsymbol{V}$, i.e., $\boldsymbol{1}_n\boldsymbol{b}_v \in \mathbb{R}^{n\times d_k}$, where $\boldsymbol{1}_n \in \mathbb{R}^{n\times 1}$ denotes an all-ones column vector. As such, $\boldsymbol{b}_v$ adds a full-dimensional linear freedom in the output space,



(a) SST-2 (b) STS-B (c) CoLA

Figure 4: Importance ranking and downstream performance of fine-tuning $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ on (a) SST-2, (b) STS-B, and (c) CoLA, with $\text{BERT}_{\text{BASE}}$. Overall, augmenting the magnitude-based approach with angular changes (Mag+Angle) shows a precise and dynamic link between bias-term rankings and downstream performance across diverse data regimes, outperforming both the Magnitude and Fisher approaches.

without being restricted by the softmax or the low-rank attention weights. In this case, $\boldsymbol{b}_v$ can be regarded as being directly added after the SDPA output; gating after SDPA and after value projection have been shown to be more effective than other positions (Qiu et al., 2025). Therefore, $\boldsymbol{b}_v$ improves the expressiveness more than $\boldsymbol{b}_q$. We note that in our discussion, we do not include the bias term of the linear layer in output projection, as $\boldsymbol{b}_v$ is able to yield an effect equivalent to that of bias terms in the output projection (Appendix-B.2) in terms of improving the expressiveness.

In summary, our discussion in this section and empirical evaluation in Section 3 show that *directly fine-tuning $\boldsymbol{b}_v$ generally leads to higher downstream performance in low-data regimes, in comparison to fine-tuning $\boldsymbol{b}_q$ or $\boldsymbol{b}_k$.*[1]

3 Evaluation

3.1 Experimental Setup

To validate our approach for effective fine-tuning, we primarily exploit $\text{BERT}_{\text{BASE}}$ (Devlin et al., 2019) on the GLUE benchmark (Wang et al., 2019b) without the WNLI task (Zaken et al., 2022). Building on this, we extend our key finding to $\text{RoBERTa}_{\text{BASE}}$ (Liu et al., 2019) and $\text{BERT}_{\text{LARGE}}$ (Devlin et al., 2019) on the GLUE benchmark and SuperGLUE benchmark (Wang et al., 2019a) (a task-specific final linear classifier layer is used for encoder-only LLMs). Furthermore, to generalize our key finding from masked LLMs to autoregressive LLMs, we exploit OPT-1.3B and OPT-6.7B (Zhang et al., 2022) evaluated on the GLUE benchmark, SuperGLUE benchmark, SQuAD (Rajpurkar et al., 2016), and DROP datasets (Dua et al., 2019), across classification, multiple-choice, and generation tasks. All these experiments undergo training on 1 NVIDIA Tesla T4 GPU with 16GB random-access memory (RAM) for $\text{BERT}_{\text{BASE}}$, $\text{RoBERTa}_{\text{BASE}}$, and $\text{BERT}_{\text{LARGE}}$, with 1 NVIDIA Tesla A40 GPU with 48GB RAM for OPT-1.3B and 1 NVIDIA Tesla A100 GPU with 80GB RAM for OPT-6.7B. The experimental details are documented in Appendix D. We also provide the paired t-test for main results to show the statistical significance in Appendix-C.2.

3.2 Bias-Term in Value vs Query and Key

We report the importance ranking and downstream performance for various bias-selection approaches, as shown in Fig. 4 and Table 1, where the importance ranking is obtained by sorting the change of bias terms, as discussed in Section 2. The greater the change, the higher the importance ranking.

In Fig. 4, we present results on three representative datasets from the GLUE benchmark with distinct evaluation indicators, and we reveal the interpretative finding across different data regimes:

[1]Note that although the investigation process of bias terms needs post-hoc evaluation, our final conclusion (w.r.t. the efficiency of directly fine-tuning $\boldsymbol{b}_v$) does not require any post-hoc evaluation.

Training Data	$b_{\mathcal{T}}$	SST-2	RTE	QQP	QNLI	MNLI$_m$	MNLI$_{mm}$	CoLA	MRPC	STS-B	Avg.
Low	b_v	**85.8**†	**59.5**†	**68.8**†	**73.8**†	**43.8**†	**45.7**†	**43.2**†	**84.0**†	**75.6**†	**64.5**
	b_q	80.0	46.9	65.1	67.7	40.1	40.5	30.8	81.1	67.8	57.8
	b_k	77.2	46.5	60.5	66.9	39.5	40.1	6.5	79.2	65.4	53.5
Medium	b_v	**87.2**†	**65.7**†	**73.5**†	**79.3**†	**58.3**†	**59.8**†	**46.3**†	**84.1**†	**83.4**†	**70.8**
	b_q	85.4	58.1	69.3	72.9	45.0	46.4	36.1	83.1	73.3	63.3
	b_k	79.7	54.5	0.00	67.6	41.4	42.1	20.3	80.8	67.1	50.4
High	b_v	**88.1**†	**62.8**†	**74.4**†	**80.8**†	**62.2**†	**64.2**†	**42.1**†	**85.5**†	**85.1**†	**71.7**
	b_q	85.8	58.4	71.8	76.6	47.8	48.5	32.9	84.2	76.9	64.7
	b_k	81.6	53.0	12.9	68.1	42.1	42.1	6.5	78.8	68.7	50.4

Table 1: Downstream performance (%) of fine-tuning different bias terms with BERT$_{\text{BASE}}$ on the GLUE benchmark. We highlight the **best** results and annotate the target bias term selected by our approach with the symbol †. For the datasets shown in Fig. 4, we additionally report the mean±std over three random seeds in Appendix C.3-Table 11.

low-, medium-, high-, and all-data regimes.

In the SST-2 dataset, as shown in Fig. 4 (a), our approach consistently selects $\boldsymbol{b}_v$ as the target bias, while the Magnitude approach always selects $\boldsymbol{b}_q$. However, the accuracy corresponding to fine-tuning $\boldsymbol{b}_v$ surpasses that of $\boldsymbol{b}_q$ across different data regimes, showing the inaccuracy of the Magnitude approach—particularly in the low-data regime. In the case of the Fisher approach, although it also identifies $\boldsymbol{b}_v$ as the target bias, it results in a static importance ranking for bias terms across different data regimes. In contrast, our approach dynamically captures the narrowing importance gap between $\boldsymbol{b}_v$ and $\boldsymbol{b}_q$ as training data increases, aligning closely with the observed convergence in their downstream performance. These results demonstrate that our approach not only achieves superior identification of target bias than the Magnitude approach, but also has the capacity of dynamic importance ranking compared to the Fisher approach (see Appendix C.1-Fig. 10 for other datasets).

Similarly, for the STS-B dataset, as shown in Fig. 4 (c), the Magnitude approach selects $\boldsymbol{b}_q$ as the target bias from medium- to all-data regimes. However, selecting $\boldsymbol{b}_q$ has a lower performance (i.e., lower Spearman correlation coefficient) than selecting $\boldsymbol{b}_v$. Although the Magnitude approach selects $\boldsymbol{b}_v$ in the low-data regime, the importance ranking gap between $\boldsymbol{b}_v$ and $\boldsymbol{b}_q$ does not align with the observed differences in downstream performance. The Fisher approach, on the other hand, suffers from the static importance ranking as before. Conversely, from low- to medium-data regimes, our approach reflects the increasing performance gap between $\boldsymbol{b}_v$ and $\boldsymbol{b}_q$, in line with their changing importance ranking.

Furthermore, for the CoLA dataset, as shown in Fig. 4 (b), our bias-efficient approach consistently selects $\boldsymbol{b}_v$ as the target bias, while the Magnitude approach always selects $\boldsymbol{b}_q$. Considering the performance metric (i.e., the Matthews correlation coefficient), the Magnitude approach selects the incorrect target bias between $\boldsymbol{b}_v$ and $\boldsymbol{b}_q$. Moreover, the Magnitude approach exhibits a static importance ranking across different data regimes. Similarly, the Fisher approach results in a fixed importance ranking. In contrast, our approach adaptively adjusts the performance gap between $\boldsymbol{b}_v$ and $\boldsymbol{b}_q$, aligning with the observed shift in their importance rankings. These results confirm the advantages of augmenting the magnitude with angular changes over Magnitude and Fisher approaches.

In Table 1, we report the downstream performance for $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$. We annotate the target bias term selected by our approach with the symbol †. In Table 1, fine-tuning $\boldsymbol{b}_v$ consistently achieves higher performance than fine-tuning $\boldsymbol{b}_q$ across the GLUE benchmark, from low- to high-data regimes. Conversely, fine-tuning $\boldsymbol{b}_k$ exhibits consistently lower performance than fine-tuning $\boldsymbol{b}_q$. As such, our approach consistently succeeds across all datasets and diverse training data regimes. These results are also consistent with the discussion in Section 2. Overall, these results show that our approach captures importance rankings and selects the target bias term to be effectively fine-tuned.

3.3 Efficiency and Effectiveness

Prior work has indicated that fine-tuning all-bias-terms has competitive performance to full-parameter fine-tuning in low-data regimes (Zaken et al., 2022; Logan IV et al., 2022; Doering et al., 2024). Accordingly, we investigate the efficiency and effectiveness of our BEFT in low-data regimes.

Fine-Tuning	Params↓	Runtime↓	Accuracy↑
Our BEFT	**0.01%**	**132.9**	58.53±1.88
Rand uniform	0.01%	659.6	50.40±2.94
All biases	0.09%	144.9	56.40±2.88
All parameters	100%	206.1	57.46±2.20

Table 2: Runtime (s) and downstream performance (%) (mean±std over three runs with different random seeds). We also highlight the second-best results, indicating the strong potential of fine-tuning the target bias term.

In Table 2, our BEFT achieves remarkable parameter efficiency with only 0.01% of the full parameters, when fine-tuning $\text{BERT}_{\text{BASE}}$ in the low-data regime of the RTE dataset. Fine-tuning all biases (Zaken et al., 2022) requires 0.09% of the full parameters, nearly 9× more trainable parameters. Although "rand uniform" samples and fine-tunes the same number of bias terms as BEFT, it yields substantially worse performance and requires more time due to inefficient uniform sampling. In terms of runtime, full-parameter fine-tuning takes 206.1 seconds (s), and all-bias fine-tuning requires 144.9 s, whereas BEFT completes in just 132.9 s. Despite this significant reduction in parameter footprint and a decrease in training time, our proposed BEFT achieves a comparable performance with all-bias fine-tuning and all-parameter fine-tuning in this case (more cases in Appendix C.1-Fig. 11). These results highlight BEFT's promise as an unprecedented parameter-efficient fine-tuning strategy.

3.4 Extension to Other Datasets and LLMs

Motivated by the preceding findings, it is natural to ask: Can the key finding (*w.r.t. the efficiency of directly fine-tuning* $\boldsymbol{b}_v$) be extended to different datasets and LLMs, and combination with PEFT methods (such as LoRA (Hu et al., 2022), VeRA (Kopiczko et al., 2024), and DoRA (Liu et al., 2024)), *without requiring any post-hoc evaluation*? The short answer is yes.

Extension to different datasets: To evaluate the extension of our key finding to different datasets in the low-data regime, we exploit the CB and WiC datasets from the SuperGLUE benchmark (Wang et al., 2019a), where the CB dataset contains only 250 training examples. First, we evaluate the same model, i.e., $\text{BERT}_{\text{BASE}}$, on these different datasets. As presented in Table 3, for $\text{BERT}_{\text{BASE}}$, fine-tuning $\boldsymbol{b}_v$ is better than fine-tuning $\boldsymbol{b}_q$; fine-tuning $\boldsymbol{b}_q$ is better than fine-tuning $\boldsymbol{b}_k$. We also extend our BEFT to multilingual and commonsense reasoning datasets in Appendix-C.7. These results consistently mirror our preceding finding on the GLUE benchmark, showing that *directly fine-tuning* $\boldsymbol{b}_v$ can generalize to new datasets.

Models	$\boldsymbol{b}_\mathcal{T}$	GLUE SST-2	GLUE RTE	SuperGLUE CB	SuperGLUE WiC
$\text{BERT}_{\text{BASE}}$	$\boldsymbol{b}_v$	**85.8**	**59.5**	**59.0**	**69.6**
	$\boldsymbol{b}_q$	80.0	46.9	47.2	66.1
	$\boldsymbol{b}_k$	77.2	46.5	43.3	62.8
$\text{RoBERTa}_{\text{BASE}}$	$\boldsymbol{b}_v$	**88.6**	**56.7**	**90.5**	**62.1**
	$\boldsymbol{b}_q$	87.0	56.3	85.0	57.8
	$\boldsymbol{b}_k$	72.0	52.3	76.2	57.9
$\text{BERT}_{\text{LARGE}}$	$\boldsymbol{b}_v$	**89.2**	**60.6**	**58.7**	**66.6**
	$\boldsymbol{b}_q$	83.8	53.4	50.9	61.2
	$\boldsymbol{b}_k$	69.3	53.4	26.6	61.2

Table 3: The key finding (*directly fine-tuning* $\boldsymbol{b}_v$) can be extended to different datasets and LLMs without requiring any post-hoc evaluation. We additionally report the mean±std over multi-seeds in Appendix C.3-Table 12.

$\boldsymbol{b}_\mathcal{T}$	Directly	+LoRA	+VeRA	+DoRA
$\boldsymbol{b}_v$	**85.8**	**85.0**	**82.2**	**86.0**
$\boldsymbol{b}_q$	80.0	82.6	81.8	83.5
$\boldsymbol{b}_k$	77.2	76.9	81.7	76.9
ΔParams↓	100%	58.3%	4.2%	59.4%

Table 4: The downstream performance (%) of variants of LoRA/VeRA/DoRA designed for BEFT on SST-2 with $\text{BERT}_{\text{BASE}}$, where $\boldsymbol{b}_v$ surpasses $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$.

Extension to different LLMs: To evaluate the extension of our key finding to different LLMs in the low-data regime, we exploit the $\text{RoBERTa}_{\text{BASE}}$ (Liu et al., 2019) and $\text{BERT}_{\text{LARGE}}$ (Devlin et al., 2019). First, on SST-2 and RTE, fine-tuning $\boldsymbol{b}_v$ consistently outperforms fine-tuning $\boldsymbol{b}_q$, as presented in Table 3. This trend holds when extending $\text{RoBERTa}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ to CB and WiC.

Extend to LoRA/VeRA/DoRA for Bias Terms: To further evaluate the extension of our key finding, we inject LoRA (Hu et al., 2022), VeRA (Kopiczko et al., 2024), and DoRA (Liu et al., 2024) into bias terms (please see Appendix-C.4), investigating the capacity of BEFT with these PEFT in low-data regimes (Schulman and Lab, 2025).

As presented in Table 4, with $(q, k, r) = (24, 32, 8)$ for $\text{BERT}_{\text{BASE}}$, by injecting LoRA or DoRA into the bias terms, more than 40% of the

LLMs	Fine-Tuning Techniques	Params	SST-2	RTE	CB	BoolQ	WSC	WiC	MultiRC	COPA	ReCoRD	SQuAD	DROP
			——————classification——————							–multiple choice–		—generation—	
OPT-1.3B	$\boldsymbol{b}_v$	0.04‰	93.1†	71.5†	**80.3**	65.8	**63.5**	**61.6**	69.0	**77.0**	**71.5**	**79.8**	**28.2**
	$\boldsymbol{b}_q$	0.04‰	72.7	55.6	71.4	60.4	62.5	59.2	56.3	74.0	71.2	69.3	21.4
	$\boldsymbol{b}_k$	0.04‰	53.5	53.1	37.5	45.3	44.2	57.0	45.3	75.0	70.5	24.2	10.4
	LoRA	1.2‰	**93.6**	**73.2**	69.6	**74.3**	**63.5**	50.0	**73.2**	76.0	71.0	79.7	27.9
	Prefix	0.4‰	91.9	63.9	**80.3**	64.3	61.5	57.6	60.0	69.0	70.5	78.3	25.2
	ICL	0‰	80.3	53.0	48.2	58.5	47.1	50.6	46.3	69.0	70.9	58.6	20.3
	Zero-Shot	0‰	53.5	53.0	39.3	45.5	44.2	57.3	45.3	75.0	70.5	27.2	11.1
OPT-6.7B	$\boldsymbol{b}_v$	0.02‰	**95.2**	**82.6**	**96.4**	78.8	**63.5**	**65.2**	73.9	**83.0**	**77.7**	**86.9**	**32.4**
	$\boldsymbol{b}_q$	0.02‰	80.3	62.4	67.8	64.7	60.5	59.4	58.6	**83.0**	76.5	75.5	27.9
	$\boldsymbol{b}_k$	0.02‰	61.2	54.8	50.0	59.5	37.5	51.2	44.5	82.0	76.0	37.4	16.8
	LoRA	0.6‰	95.1	81.9	75.0	**79.1**	**63.5**	50.4	**74.4**	82.0	76.3	85.7	31.1
	Prefix	0.2‰	95.1	78.3	89.2	72.6	59.6	54.7	58.9	78.0	76.8	84.6	30.5
	ICL	0‰	84.6	65.7	57.1	68.9	50.9	53.6	50.4	82.0	76.8	74.2	27.5
	Zero-Shot	0‰	61.2	55.2	51.7	59.5	37.5	51.2	44.5	82.0	76.1	36.5	17.7

Table 5: Downstream performance (%) of fine-tuning techniques in low-data regimes. For our key finding, we additionally report the mean±std over three random seeds in Appendix C.3-Table 10. BEFT demonstrates competitive performance with mainstream PEFT techniques regarding parameter-efficiency and downstream adaptation.

trainable parameters can be reduced, while over 95% for VeRA. At the same time, the downstream performance of fine-tuning $\boldsymbol{b}_v$ with these PEFT methods consistently outperforms fine-tuning $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$. The key finding can also be directly combined with LoRA/VeRA/DoRA (more variants and LLMs in Appendix-C.4 Table 13 and 14). Moreover, we also extend our BEFT to more advanced PEFT, such as PiSSA (Meng et al., 2024) and BA-LoRA (Chang et al., 2026), in Appendix-C.5.

Overall, these results show that *directly fine-tuning* $\boldsymbol{b}_v$ can be effective to different datasets and LLMs, and be combined with LoRA, VeRA, and DoRA, without requiring any post-hoc evaluation.

3.5 Extension to Autoregressive LLMs

To further validate the generality of our finding from masked LLMs to autoregressive LLMs, we consider OPT-1.3B and OPT-6.7B (Zhang et al., 2022) on various benchmarks, including the GLUE (Wang et al., 2019b), SuperGLUE (Wang et al., 2019a), SQuAD (Rajpurkar et al., 2016), and DROP datasets (Dua et al., 2019) covering classification, multiple-choice, and generation tasks. We also compare our BEFT against various mainstream PEFT methods, such as LoRA (Hu et al., 2022), prefix tuning (Li and Liang, 2021), in-context learning (ICL) (Brown et al., 2020) and zero-shot techniques (Brown et al., 2020), as presented in Table 5. First, fine-tuning $\boldsymbol{b}_v$ consistently performs the best among $\boldsymbol{b}_v$, $\boldsymbol{b}_q$, and $\boldsymbol{b}_k$, which demonstrates the effective generalization of our key finding (for instance, the results of SST-2 with OPT-1.3B match the target bias in Appendix C.1-Fig. 12). Second, our BEFT requires 30× and 10× fewer parameters than LoRA and prefix tuning, respectively, while maintaining competitive or superior performance across tasks. In addition, our BEFT also demonstrates the competence when compared to more advanced techniques such as VeRA (Kopiczko et al., 2024) and DoRA (Liu et al., 2024) in Appendix-C.6. Finally, while ICL and zero-shot techniques do not require fine-tuning, they exhibit inherently inferior performance compared to our BEFT. Overall, these results indicate that our BEFT approach achieves competitive performance and efficiency relative to the mainstream and more advanced PEFT techniques.

LLaMA2-7B	Params	SST-2	COPA	SQuAD
Adding $\boldsymbol{b}_v$	0.02‰	**94.9**±0.7	**85.0**±0.0	**89.4**±0.8
Adding $\boldsymbol{b}_q$	0.02‰	90.0±0.3	79.0±0.0	88.0±0.6
Adding $\boldsymbol{b}_k$	0.02‰	68.0±0.4	79.0±0.0	82.0±0.5
LoRA	0.6‰	95.6±0.1	83.3±0.5	90.4±0.6
Prefix	0.2‰	93.5±0.4	79.0±0.0	89.9±1.8

Table 6: Downstream performance (%) (mean±std over three runs with different random seeds) of adding bias into bias-free LLMs, where our key finding still holds.

3.6 Extension to Bias-Free LLMs

Today, certain LLMs are bias-free, consequently omitting bias terms in their attention modules (Touvron et al., 2023; Guo et al., 2024; Team et al., 2024). Nevertheless, methods such as per-layer bias-terms injection (Li et al., 2023), scaling and biasing operation (Wu et al., 2024a,b), and acti-

vating only layer-wise additive biases (Sinii et al., 2025) have shown both effectiveness and efficiency. To validate our key finding with bias-free LLMs, we manually add different bias terms to their attention module. As shown in Table 6, we add $\boldsymbol{b}_v$, $\boldsymbol{b}_q$, and $\boldsymbol{b}_k$ into LLaMA (LLaMA-2-7B-hf) (Touvron et al., 2023). The downstream performance of fine-tuning additive $\boldsymbol{b}_v$ still surpasses that of fine-tuning additive $\boldsymbol{b}_q$ and additive $\boldsymbol{b}_k$. Similar patterns are observed and presented for GPT-J-6B (Wang and Komatsuzaki, 2021) and DeepSeek-Coder-Base-1.3B (Guo et al., 2024) in Appendix-C.8. These results show that for bias-free LLMs in the low-data regime, adding $\boldsymbol{b}_v$ and fine-tuning $\boldsymbol{b}_v$ can lead to competitive performance, without requiring any post-hoc evaluation. In addition, adding $\boldsymbol{b}_v$ requires $30\times$ and $10\times$ fewer parameters than LoRA and prefix tuning, respectively, while maintaining competitive performance across tasks.

4 Related Work

4.1 Parameter-Efficient Fine-Tuning (PEFT)

PEFT techniques aim to fine-tune only a small subset of parameters in LLMs, thereby significantly reducing resource overheads. Addition-based PEFT methods introduce new trainable components into the model architectures. Prefix tuning (Li and Liang, 2021) adds trainable prefix tokens to attention blocks; adapter tuning (Houlsby et al., 2019) inserts a trainable adapter module into Transformer layers; and prompt-tuning (Lester et al., 2021) introduces a sequence of trainable embeddings prepended to the input. Reparameterization-based PEFT methods change the representation of the parameter or its update. LoRA (Hu et al., 2022) exploits the low-rank decomposition to approximate the original parameter change, and GaLore (Zhao et al., 2024) reparameterizes the gradient into low-rank matrices. While effective, these PEFT techniques are limited by their out-of-the-box usability, with extra requirements for additional configuration and auxiliary reparameterization.

4.2 Bias-Efficient Fine-Tuning

Bias-efficient fine-tuning techniques aim to fine-tune only the bias terms in LLMs. Fine-tuning all biases—as in BitFit (Zaken et al., 2022)—has been shown to achieve competitive performance with full-parameter fine-tuning, especially in low-data regimes (Zaken et al., 2022; Logan IV et al., 2022). This advantage is attributed in part to the low intrinsic dimensionality of pre-trained LLMs, which allows meaningful adaptation within a restricted subspace (Aghajanyan et al., 2021; Mosbach et al., 2021; Tang et al., 2024). Prior papers reveal that different bias terms contribute differently during model adaptation (Ding et al., 2023; Weng et al., 2024). However, the link between fine-tuning different bias terms and downstream performance during model adaptation is not clear (Ding et al., 2023; Weng et al., 2024). Moreover, the selective PEFT approaches, such as the magnitude of bias change (Ansell et al., 2022) and empirical Fisher information (Sung et al., 2021; Xue et al., 2025; Guo et al., 2025), provide limited guidance to select the target bias term. As a result, strategies for selecting target bias terms for effective fine-tuning remain largely unexplored.

5 Conclusions

Bias-only fine-tuning of LLMs provides out-of-the-box usability, and demonstrates competitive performance compared to full-parameter fine-tuning, particularly in low-data regimes (Zaken et al., 2022; Logan IV et al., 2022; Doering et al., 2024). In this paper, we introduce a simple yet effective approach, based on the potential expressive power of bias terms in query, key, and value projections, to investigate the target bias term for effective fine-tuning. We extensively evaluate our approach and shed light on the link between fine-tuning different bias terms and downstream performance, across diverse data regimes. Our key finding is that directly fine-tuning $\boldsymbol{b}_v$ generally leads to higher downstream performance in low-data regimes, compared to $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$, without requiring any post-hoc evaluation. Moreover, we extend and generalize our key finding to various LLMs covering encoder-only (masked) and decoder-only (autoregressive) LLMs (Vaswani et al., 2017) with sizes ranging from 110M to 6.7B parameters, as well as bias-free LLMs, across various downstream tasks, including classification, multiple-choice, and generation tasks. At the same time, our work is compatible and can be readily combined with PEFT methods, e.g., LoRA, VeRA, DoRA, and PiSSA. Our results show the effectiveness of directly fine-tuning $\boldsymbol{b}_v$, without any post-hoc evaluation.

Limitations

In this work, we shed light on the importance of fine-tuning bias terms in LLMs for unprecedented

parameter efficiency. The limitations could be summarized as follows: (1) Our paper mainly focuses on standard softmax attention, not including linear attention (Katharopoulos et al., 2020; Choromanski et al., 2021); (2) Our paper mainly focuses on the query, key, and value projections for SDPA, similar to previous work (Qiu et al., 2025). Recent studies have shown that LayerNorm can potentially be replaced by a simple dynamic Tanh function (Zhu et al., 2025) and even further by a rescaled Gaussian cumulative distribution function (Chen et al., 2025). At the same time, the feed-forward network can potentially be replaced by efficient non-linear mapping into low-dimensional computation (Kim et al., 2025); (3) Our paper analyzes the improvement in expressiveness brought by $\boldsymbol{b}_q$, $\boldsymbol{b}_k$, and $\boldsymbol{b}_v$ in Section 2. How these bias terms influence the degree of superposition (Liu et al., 2025) and, consequently, the scaling law (Kaplan et al., 2020), is left for future investigation.

Acknowledgments

This research has been partially supported by the Swedish Wallenberg AI, Autonomous Systems and Software Program (WASP), Swedish Research Council, Swedish Foundation for Strategic Research, ELLIIT Strategic Research Environment, and an unrestricted gift from Google.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328.

Alan Ansell, Edoardo Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1778–1796.

Massimo Bini, Leander Girrbach, and Zeynep Akata. 2025. Decoupling angles and strength in low-rank adaptation. In *The Thirteenth International Conference on Learning Representations*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yupeng Chang, Yi Chang, and Yuan Wu. 2026. BA-loRA: Bias-alleviating low-rank adaptation to mitigate catastrophic inheritance in large language models. In *The Fourteenth International Conference on Learning Representations*.

Mingzhi Chen, Taiming Lu, Jiachen Zhu, Mingjie Sun, and Zhuang Liu. 2025. Stronger normalization-free transformers. *Preprint*, arXiv:2512.10938.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. 2021. Rethinking attention with performers. In *International Conference on Learning Representations*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, and 1 others. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Nigel Doering, Cyril Gorlla, Trevor Tuttle, and Adhvaith Vijay. 2024. Empirical analysis of efficient fine-tuning methods for large pre-trained language models. *arXiv preprint arXiv:2401.04051*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, and 1 others. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

Wentao Guo, Jikai Long, Yimeng Zeng, Zirui Liu, Xinyu Yang, Yide Ran, Jacob R Gardner, Osbert Bastani, Christopher De Sa, Xiaodong Yu, and 1 others. 2025. Zeroth-order fine-tuning of llms with transferable static sparsity. In *The Thirteenth International Conference on Learning Representations*.

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

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR.

Sang Min Kim, Byeongchan Kim, Arijit Sehanobish, Somnath Basu Roy Chowdhury, Rahul Kidambi, Dongseok Shim, Kumar Avinava Dubey, Snigdha Chaturvedi, Min hwan Oh, and Krzysztof Marcin Choromanski. 2025. EUGens: Efficient, unified and general dense layers. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. 2024. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*.

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

Yizhou Liu, Ziming Liu, and Jeff Gore. 2025. Superposition yields robust neural scaling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Robert Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2022. Cutting down on prompts and parameters: Simple few-shot learning with language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2824–2835.

Martin Marek, Sanae Lotfi, Aditya Somasundaram, Andrew Gordon Wilson, and Micah Goldblum. 2025. Small batch size training for language models: When vanilla SGD works, and why gradient accumulation is wasteful. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

John Schulman and Thinking Machines Lab. 2025. Lora without regret. *Thinking Machines Lab: Connectionism*. Https://thinkingmachines.ai/blog/lora/.

Viacheslav Sinii, Alexey Gorbatovski, Artem Cherepanov, Boris Shaposhnikov, Nikita Balagansky, and Daniil Gavrilov. 2025. Steering LLM reasoning through bias-only adaptation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9213–9222, Suzhou, China. Association for Computational Linguistics.

Yi-Lin Sung, Varun Nair, and Colin A Raffel. 2021. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34:24193–24205.

Yuwei Tang, Zhenyi Lin, Qilong Wang, Pengfei Zhu, and Qinghua Hu. 2024. Amu-tuning: Effective logit bias for clip-based few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23323–23333.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Yan Roger Weng, Sean Kerrigan, and Kevin Yang. 2024. Bitfit+: Fine-tuning bias and gamma parameters.

Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Zhu JianHao, Cenyuan Zhang, Xiaoqing Zheng, and Xuan-Jing Huang. 2024a. Advancing parameter efficiency in fine-tuning via representation editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13445–13464.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. 2024b. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

Kang Xue, Ming Dong, Xinhui Tu, and Tingting He. 2025. Fish-tuning: Enhancing peft methods with fisher information. *arXiv preprint arXiv:2504.04050*.

Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Shaochen Zhong, Bing Yin, and Xia Hu. 2024. Harnessing the power of llms in practice: A survey on chatgpt and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6):1–32.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. 2024. Galore: Memory-efficient LLM training by gradient low-rank projection. In *Forty-first International Conference on Machine Learning*.

Jiachen Zhu, Xinlei Chen, Kaiming He, Yann LeCun, and Zhuang Liu. 2025. Transformers without normalization. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 14901–14911.

Appendix

Contents

A Different Approaches of Investigating Bias Terms

A.1 Magnitude Approach

We denote the bias term at layer l before fine-tuning as $\boldsymbol{b}_{\mathcal{T}}^{(l),pre}$ and after fine-tuning as $\boldsymbol{b}_{\mathcal{T}}^{(l),post}$. The change of the bias term across all layers in Magnitude approaches (Zaken et al., 2022) is denoted as:

$$\Delta(\boldsymbol{b}_{\mathcal{T}}) = \frac{1}{L}\sum_{l=1}^{L}\left\|\boldsymbol{b}_{\mathcal{T}}^{(l),post} - \boldsymbol{b}_{\mathcal{T}}^{(l),pre}\right\|_1.$$

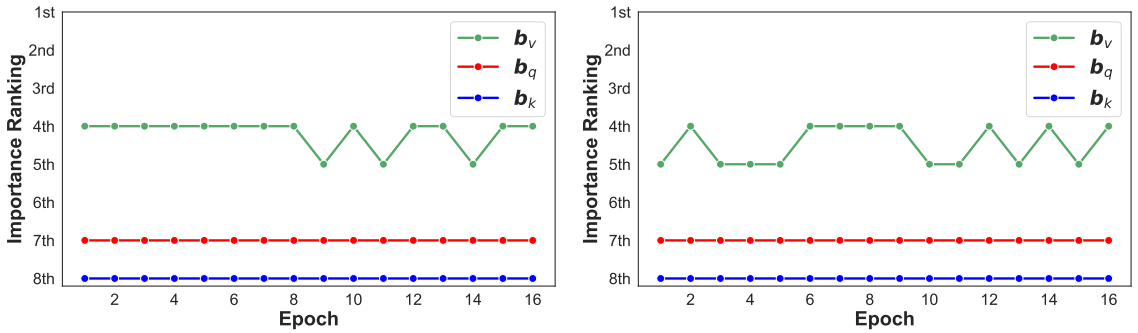
A.2 Fisher Approach

For the Fisher information, the effect of changes in the bias before fine-tuning ($\boldsymbol{b}_{\mathcal{T}}^{pre}$) on the output $\boldsymbol{y}$ is measured, given training data $\boldsymbol{x}$. Fisher information is typically approximated using its diagonal for computational efficiency, giving rise to the empirical Fisher information applied in supervised learning.

For a LLM model with pre-trained parameters $\boldsymbol{\theta} \in \mathbb{R}^{|\boldsymbol{\theta}|}$, the output $\boldsymbol{y}$ is inferred by the input $\boldsymbol{x}$ and the parameters $\boldsymbol{\theta}$, i.e., $\boldsymbol{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x})$. To measure how much the prediction would change for a given change in parameters $\boldsymbol{\theta}$, the Fisher information $F_{\boldsymbol{\theta}} \in \mathbb{R}^{|\boldsymbol{\theta}|\times|\boldsymbol{\theta}|}$ is introduced as below, $F_{\boldsymbol{\theta}} = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\left[\mathbb{E}_{\boldsymbol{y}\sim p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})}\nabla_{\boldsymbol{\theta}}\log p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})\nabla_{\boldsymbol{\theta}}\log p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})^T\right]$, where $p(\boldsymbol{x})$ is the probability distribution of the input $\boldsymbol{x}$ and $p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})$ is the probability distribution of the output $\boldsymbol{y}$ based on the LLM model parameters $\boldsymbol{\theta}$ and input $\boldsymbol{x}$. $\nabla\log p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})$ is the gradient of the log-likelihood with respect to θ. In LLMs, $|\boldsymbol{\theta}| \times |\boldsymbol{\theta}|$ has heavy computation, making it impossible to compute. Therefore, the diagonal vector is exploited as the approximated Fisher information. Furthermore, due to the limited training data, the approximated Fisher information $\hat{F}_{\boldsymbol{\theta}} \in \mathbb{R}^{|\boldsymbol{\theta}|}$ is introduced as below:

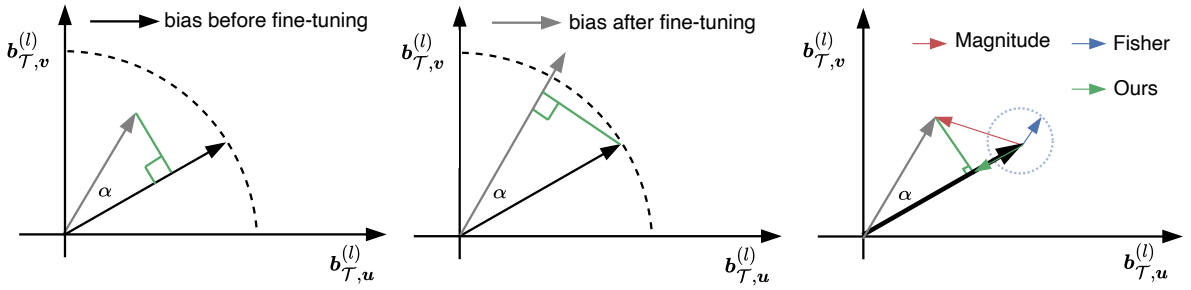
$$\hat{F}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{\boldsymbol{y}\sim p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}_i)}(\nabla_{\boldsymbol{\theta}}\log p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x}_i))^2,$$

where N is the total number of training data. Moreover, in supervised learning, we can use empirical



(a) Batch size 16 (b) Batch size 8

Figure 5: Fisher Approach: importance ranking of different bias terms when fine-tuning $\text{BERT}_{\text{BASE}}$ on the SST2 low-data regime as epoch increasing. Although different batch sizes introduce slight variations to the importance ranking of $\boldsymbol{b}_v$, the underlying pattern (mostly static) remains consistent.



(a) $\boldsymbol{b}_{\mathcal{T}}^{(l),post} < \boldsymbol{b}_{\mathcal{T}}^{(l),pre}$ (b) $\boldsymbol{b}_{\mathcal{T}}^{(l),post} > \boldsymbol{b}_{\mathcal{T}}^{(l),pre}$ (c) Comparison

Figure 6: We propose a simple yet effective approach, incorporating angular changes α into magnitude changes, to measure the change of $\boldsymbol{b}_{\mathcal{T}}^{(l)}$ by projection and scaling (simplified visualization in $(\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{R}^2$).

Fisher information as below:

$$\hat{F}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}(\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{y}_i|\boldsymbol{x}_i))^2.$$

In supervised learning, this gradient $\nabla_\theta \log p_{\boldsymbol{\theta}}(\boldsymbol{y}_i|\boldsymbol{x}_i)$ is simplified as $-\nabla_\theta \mathcal{L}(\boldsymbol{y}_i, f_{\boldsymbol{\theta}}(\boldsymbol{x}_i))$, where $\mathcal{L}$ is the Cross-Entropy loss.

Therefore, the change of the bias term across all layers in the Fisher approach is denoted as follows:

$$\Delta(\boldsymbol{b}_{\mathcal{T}}) = \frac{1}{L \cdot N}\sum_{l=1}^{L}\sum_{i=1}^{N}\left(\nabla_{\boldsymbol{b}_{\mathcal{T}}^{(l),pre}} \log p_{\theta}(\boldsymbol{y}_i|\boldsymbol{x}_i)\right)^2,$$

where N is the total number of training data. $p_{\boldsymbol{\theta}}(\boldsymbol{y}_i|\boldsymbol{x}_i)$ is the probability distribution of the output $\boldsymbol{y}_i$ based on the LLM model parameters $\boldsymbol{\theta}$ and input $\boldsymbol{x}_i$. $\nabla_{\boldsymbol{b}_{\mathcal{T}}^{(l),pre}} \log p_{\boldsymbol{\theta}}(\boldsymbol{y}_i|\boldsymbol{x}_i)$ is the gradient of the log-likelihood with respect to the bias $\boldsymbol{b}^{(l),pre}$.

A.3 Our Approach

When the magnitude of fine-tuned bias is smaller than the magnitude of the bias before fine-tuning, as illustrated in Fig. 6 (a), the projection of fine-tuned bias onto the bias before fine-tuning is considered to calculate the $\Delta(\boldsymbol{b}_{\mathcal{T}}^{(l)})$:

$$\Delta(\boldsymbol{b}_{\mathcal{T}}^{(l)}) = 1 - \frac{\|\boldsymbol{b}_{\mathcal{T}}^{(l),post}\|_2 \cdot \cos(\alpha)}{\|\boldsymbol{b}_{\mathcal{T}}^{(l),pre}\|_2} = 1 - \frac{\boldsymbol{b}_{\mathcal{T}}^{(l),pre} \cdot \boldsymbol{b}_{\mathcal{T}}^{(l),post}}{\|\boldsymbol{b}_{\mathcal{T}}^{(l),pre}\|_2^2},$$

where α is the angle between the bias before and after fine-tuning. On the contrary, when the magnitude of fine-tuned bias is not smaller than the magnitude of the bias before fine-tuning, as illustrated in Fig. 6 (b), the projection of the bias before fine-tuning onto the fine-tuned bias is considered to calculate the $\Delta(\boldsymbol{b}_{\mathcal{T}}^{(l)})$:

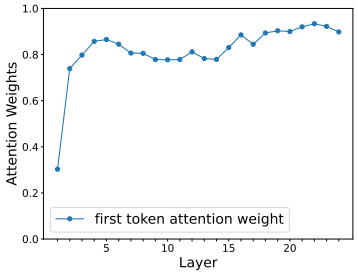
$$\Delta(\boldsymbol{b}_{\mathcal{T}}^{(l)}) = 1 - \frac{\|\boldsymbol{b}_{\mathcal{T}}^{(l),pre}\|_2 \cdot \cos(\alpha)}{\|\boldsymbol{b}_{\mathcal{T}}^{(l),post}\|_2} = 1 - \frac{\boldsymbol{b}_{\mathcal{T}}^{(l),pre} \cdot \boldsymbol{b}_{\mathcal{T}}^{(l),post}}{\|\boldsymbol{b}_{\mathcal{T}}^{(l),post}\|_2^2}.$$

To present a unified formulation considering all layers, we have:

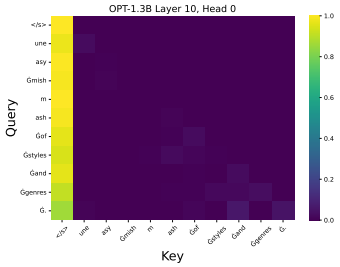
$$\Delta(\boldsymbol{b}_{\mathcal{T}}) = \frac{1}{L}\sum_{l=1}^{L}\left(1 - \frac{\boldsymbol{b}_{\mathcal{T}}^{(l),pre} \cdot \boldsymbol{b}_{\mathcal{T}}^{(l),post}}{\max\left(\|\boldsymbol{b}_{\mathcal{T}}^{(l),pre}\|_2^2, \|\boldsymbol{b}_{\mathcal{T}}^{(l),post}\|_2^2\right)}\right).$$

B Extension of Analysis

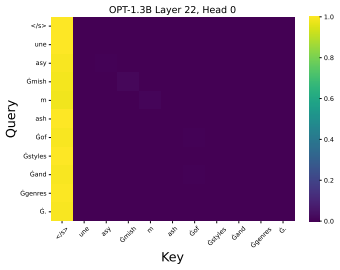
B.1 Inherent Sharpness and High Sparsity of Attention Weights



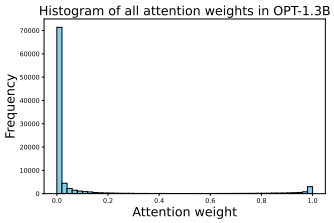
(a) Proportion of attention allocated to the first token per layer (OPT-1.3B).



(b) Attention weights for one head of the 10th layer (OPT-1.3B).

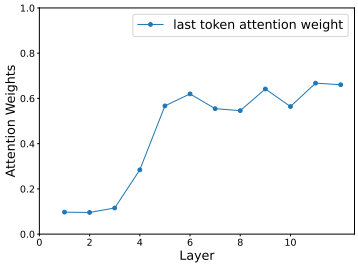


(c) Attention weights for one head of the 22nd layer (OPT-1.3B).

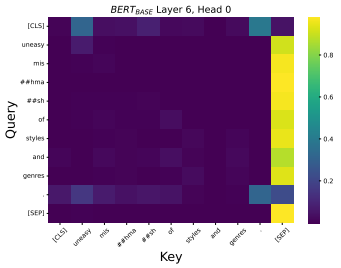


(d) The histograms of all attention weights (OPT-1.3B).

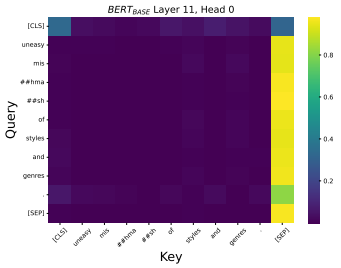
Figure 7: The observed attention sink (Xiao et al., 2024; Qiu et al., 2025) and the histograms of all attention weights for OPT-1.3B (unidirectional attention) with one test input from the SST-2—"*uneasy mishmash of styles and genres .*". In OPT-1.3B, attention weights across layers tend to be directed towards the first token and show high sparsity.



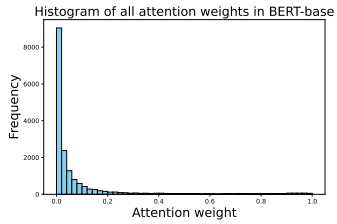
(a) Proportion of attention allocated to the last token per layer ($\text{BERT}_{\text{BASE}}$).



(b) Attention weights for one head of the 6th layer ($\text{BERT}_{\text{BASE}}$).



(c) Attention weights for one head of the 11st layer ($\text{BERT}_{\text{BASE}}$).



(d) The histograms of all attention weights ($\text{BERT}_{\text{BASE}}$).

Figure 8: Similar attention sink and the histograms of all attention weights for $\text{BERT}_{\text{BASE}}$ (bidirectional attention) with one test input from the SST-2—"*uneasy mishmash of styles and genres .*". In $\text{BERT}_{\text{BASE}}$, the attention sink happens for the last token and show high sparsity.

B.2 Equivalence of Bias Terms in the Value Projection and Output Projection

Based on Equation (1), if only $\boldsymbol{b}_v$ available, for the whole attention weights $\boldsymbol{A}$ of each attention head, we have:

$$\text{head}^i = \boldsymbol{A}^i(\boldsymbol{V}^i + \boldsymbol{b}_v^i) = \boldsymbol{A}^i\boldsymbol{V}^i + \boldsymbol{b}_v^i, \tag{2}$$

where i is the index of attention head; $\boldsymbol{A}^i\boldsymbol{V}^i \in \mathbb{R}^{n\times d_k}$ and $\boldsymbol{b}_v^i \in \mathbb{R}^{1\times d_k}$. $\boldsymbol{b}_v^i$ is broadcast across the rows of $\boldsymbol{A}^i\boldsymbol{V}^i$. For multi-head attention, Equation (2) is repeated in parallel for h heads and $h = \frac{d_{\text{model}}}{d_k}$, where d_{model} is the model dimension and d_k is the dimension of one head.

The outputs of SDPA are concatenated as:

$$\text{MultiHead} = \text{Concat}(\text{head}^1, ..., \text{head}^h),$$

where the $\text{MultiHead} \in \mathbb{R}^{n\times d_{\text{model}}}$ is passed through the output projection as shown in Fig. 9. For the output projection, we have $\boldsymbol{W}_o \in \mathbb{R}^{d_{\text{model}}\times d_{\text{model}}}$ and $\boldsymbol{b}_o \in \mathbb{R}^{1\times d_{\text{model}}}$. If only $\boldsymbol{b}_v$ available, the calculation is:

$$\begin{aligned}
\text{MultiHead}_{(\text{with } \boldsymbol{b}_v)}\boldsymbol{W}_o &= \text{Concat}(\text{head}^1, ..., \text{head}^h)\boldsymbol{W}_o \\
&= \text{Concat}(\boldsymbol{A}^1\boldsymbol{V}^1 + \boldsymbol{b}_v^1, ..., \boldsymbol{A}^h\boldsymbol{V}^h + \boldsymbol{b}_v^h)\boldsymbol{W}_o \\
&= \left[\text{Concat}(\boldsymbol{A}^1\boldsymbol{V}^1, ..., \boldsymbol{A}^h\boldsymbol{V}^h) + \text{Concat}(\boldsymbol{b}_v^1, ..., \boldsymbol{b}_v^h)\right]\boldsymbol{W}_o \\
&= \text{Concat}(\boldsymbol{A}^1\boldsymbol{V}^1, ..., \boldsymbol{A}^h\boldsymbol{V}^h)\boldsymbol{W}_o + \text{Concat}(\boldsymbol{b}_v^1, ..., \boldsymbol{b}_v^h)\boldsymbol{W}_o,
\end{aligned} \tag{3}$$

where $\text{Concat}(\boldsymbol{A}^1\boldsymbol{V}^1, ..., \boldsymbol{A}^h\boldsymbol{V}^h) \in \mathbb{R}^{n\times d_{\text{model}}}$ and $\text{Concat}(\boldsymbol{b}_v^1, ..., \boldsymbol{b}_v^h) \in \mathbb{R}^{1\times d_{\text{model}}}$. $\text{Concat}(\boldsymbol{b}_v^1, ..., \boldsymbol{b}_v^h)$ is broadcast across the rows of $\text{Concat}(\boldsymbol{A}^1\boldsymbol{V}^1, ..., \boldsymbol{A}^h\boldsymbol{V}^h)$.

If only $\boldsymbol{b}_o$ available, the calculation is:

$$\begin{aligned}
\text{MultiHead}_{(\text{without } \boldsymbol{b}_v)}\boldsymbol{W}_o + \boldsymbol{b}_o &= \text{Concat}(\text{head}^1, ..., \text{head}^h)\boldsymbol{W}_o + \boldsymbol{b}_o \\
&= \text{Concat}(\boldsymbol{A}^1\boldsymbol{V}^1, ..., \boldsymbol{A}^h\boldsymbol{V}^h)\boldsymbol{W}_o + \boldsymbol{b}_o.
\end{aligned} \tag{4}$$

Based on Equation (3) and Equation (4), let $\boldsymbol{b}_{o,j}$ denote the j-th elements of $\boldsymbol{b}_o$ and $\boldsymbol{W}_{o,j}$ denote the j-th column of $\boldsymbol{W}_o$. When

$$\boldsymbol{b}_{o,j} = \text{Concat}(\boldsymbol{b}_v^1, ..., \boldsymbol{b}_v^h)\boldsymbol{W}_{o,j} \quad \forall j \in \{1, ..., d_{\text{model}}\},$$

we can get

$$\text{MultiHead}_{(\text{with } \boldsymbol{b}_v)}\boldsymbol{W}_o = \text{MultiHead}_{(\text{without } \boldsymbol{b}_v)}\boldsymbol{W}_o + \boldsymbol{b}_o.$$

This derivation demonstrates the equivalence of bias terms in the value projection and output projection. In terms of improving the expressiveness, $\boldsymbol{b}_v$ is able to yield an effect equivalent to that of $\boldsymbol{b}_o$. We also empirically fine-tuning $\boldsymbol{b}_v$ and $\boldsymbol{b}_o$, and validate the claimed equivalence, as presented in Table 7.

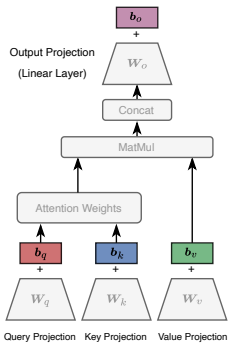


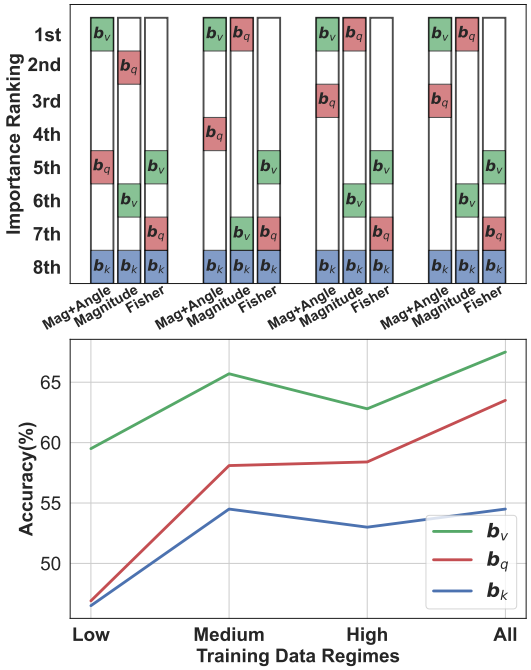
Figure 9: The SDPA with output projection, where the output projection is a linear layer.

Multi-Seed	$\boldsymbol{b}_v$	$\boldsymbol{b}_o$
0	93.1	93.1
1	93.6	92.8
2	93.6	93.7
3	92.4	92.3
4	93.2	93.1
mean ± std	93.1±0.44	93.0±0.45

Table 7: Downstream performance (%) of fine-tuning $\boldsymbol{b}_v$ and $\boldsymbol{b}_o$ on SST-2 dataset with OPT-1.3B.

C Investigation of Extensions

C.1 The Effectiveness of Our Key Finding



(a) RTE

Figure 10: Importance ranking and downstream performance of fine-tuning different bias terms on the RTE dataset with $\text{BERT}_{\text{BASE}}$. Still, our approach demonstrates the ability to precisely and dynamically identify the target bias term across low-data to high-data regimes, outperforming both Magnitude and Fisher approaches.

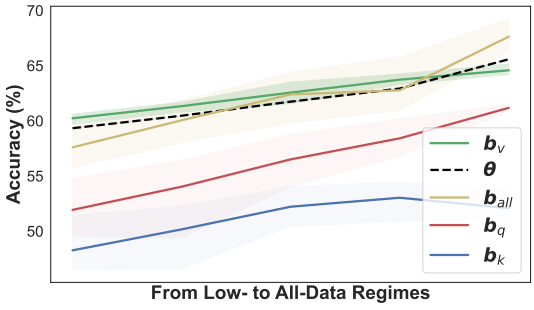


Figure 11: Fine-tuning various subsets of LLM parameters. The solid line reports the mean over three different seeds, and the shaded area indicates the standard deviation. We report the last epoch accuracy on validation dataset for consistent comparison, while the BitFit (Zaken et al., 2022) reports the best validation accuracy across training for fine-tuning $\boldsymbol{b}_{all}$.

C.2 Statistical Significance

We did the paired t-test for the main results in Table 1 and Table 5. For Table 1, we pick CoLA and STS-B to do paired t-test, where the results are statistically significant (p<0.05), as shown below. To further investigate this paired t-test, taking SST-2 for example, we repeat the experiments 5 times with different random seeds to obtain the p-values of 0.003 and 0.013, showing the statistical significance of these results. When we repeat the experiments 10 times with different random seeds, we obtain the p-values of 1.44×10^{-6} and 2.78×10^{-5}, showing the high statistical significance.

For Table 5, we pick OPT-1.3B on CB, SQuAD, and DROP, where the results are statistically significant (p<0.05), as shown below:

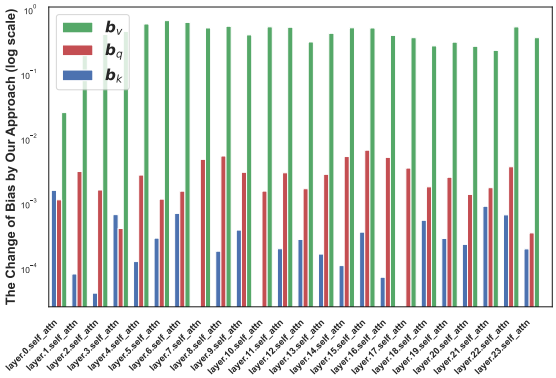


Figure 12: The change of bias (log scale), across Transformer layers, by our approach with OPT-1.3B on SST-2. The target bias is $\boldsymbol{b}_v$.

	CoLA	STS-B
Fine-Tuning $\boldsymbol{b}_v$	40.2±2.85	75.3±0.65
t-test ($\boldsymbol{b}_v$ vs $\boldsymbol{b}_q$)	p=0.005	p= 0.001
Fine-Tuning $\boldsymbol{b}_q$	27.5±3.98	67.5±0.28
t-test ($\boldsymbol{b}_q$ vs $\boldsymbol{b}_k$)	p=0.003	p=0.026
Fine-Tuning $\boldsymbol{b}_k$	4.1±2.93	65.6±0.48

Table 8: Paired t-test shows the statistical significance of our results in Table 1.

	CB	SQuAD	DROP
Fine-Tuning $\boldsymbol{b}_v$	77.3±2.2	78.6±0.9	28.8±0.5
t-test ($\boldsymbol{b}_v$ vs $\boldsymbol{b}_q$)	p=0.023	p= 0.0004	p=0.021
Fine-Tuning $\boldsymbol{b}_q$	70.2±1.7	68.5±0.6	22.4±1.7
t-test ($\boldsymbol{b}_q$ vs $\boldsymbol{b}_k$)	p=0.001	p=0.0001	p=0.009
Fine-Tuning $\boldsymbol{b}_k$	37.5±0.0	24.6±0.3	10.3±0.1

Table 9: Paired t-test shows the statistical significance of our results in Table 5.

LLMs	Fine-Tuning Techniques	SST-2	RTE	CB	BoolQ	WSC	WiC	MultiRC	COPA	ReCoRD	SQuAD	DROP
		——————————classification——————————							–multiple choice–		—generation—	
OPT-1.3B	$\boldsymbol{b}_v$	**93.4**±0.2	**70.5**±1.1	**77.3**±2.2	**66.4**±0.9	**63.4**±0.1	**61.8**±1.2	**63.8**±3.8	**77.6**±0.5	**71.2**±0.7	**78.6**±0.9	**28.8**±0.5
	$\boldsymbol{b}_q$	73.5±0.7	56.4±1.0	70.2±1.7	60.8±1.1	62.8±1.2	58.9±0.2	56.6±0.4	74.0±0.0	70.6±0.6	68.5±0.6	22.4±1.7
	$\boldsymbol{b}_k$	53.5±0.0	53.0±0.1	37.5±0.0	46.8±1.6	44.2±0.0	57.0±0.0	45.8±1.2	75.0±0.0	70.0±0.3	24.6±0.3	10.3±0.1
OPT-6.7B	$\boldsymbol{b}_v$	**95.4**±0.3	**81.4**±0.8	**82.1**±10.5	**79.1**±0.3	**63.5**±0.0	**64.6**±2.1	**73.2**±1.5	**83.0**±0.0	**78.2**±0.7	**85.6**±0.9	**32.8**±0.6
	$\boldsymbol{b}_q$	80.8±0.4	63.4±1.0	67.8±1.5	65.2±0.4	61.5±0.8	59.3±0.2	58.1±1.8	**83.0**±0.0	77.0±0.5	74.6±0.6	27.0±0.6
	$\boldsymbol{b}_k$	61.2±0.0	54.8±0.0	50.0±0.0	61.2±1.2	37.5±0.0	51.2±0.0	44.5±0.7	82.0±0.0	76.4±0.6	38.0±0.9	15.6±0.8

Table 10: Downstream performance (%) (mean±std over three runs with different random seeds).

C.3 Stability Across Random Seeds

$\boldsymbol{b}_\mathcal{T}$	SST-2	CoLA	STS-B
$\boldsymbol{b}_v$	**83.9**±1.39	**40.2**±2.85	**75.3**±0.65
$\boldsymbol{b}_q$	75.6±4.36	27.5±3.98	67.5±0.28
$\boldsymbol{b}_k$	72.6±4.61	4.1±2.93	65.6±0.48

Table 11: Downstream performance (%) (mean±std over three runs with different random seeds) of fine-tuning different bias terms on representative tasks as presented in Fig. 4, with $\text{BERT}_{\text{BASE}}$ in low-data regime.

Models	$\boldsymbol{b}_\mathcal{T}$	SST-2	WiC
$\text{RoBERTa}_{\text{BASE}}$	$\boldsymbol{b}_v$	**89.6**±0.72	**61.8**±1.12
	$\boldsymbol{b}_q$	84.7±3.30	56.5±0.94
	$\boldsymbol{b}_k$	72.6±7.84	56.6±0.95
$\text{BERT}_{\text{LARGE}}$	$\boldsymbol{b}_v$	**89.8**±0.58	**67.3**±1.03
	$\boldsymbol{b}_q$	83.6±0.33	62.6±1.06
	$\boldsymbol{b}_k$	73.2±3.60	61.5±0.51

Table 12: Downstream performance (%) (mean±std over three runs with different random seeds) on SST-2 (GLUE) and WiC (SuperGLUE), with $\text{RoBERTa}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ in low-data regime.

C.4 LoRA/VeRA/DoRA for Bias Terms

For a pre-trained bias vector $\boldsymbol{b}_{\mathcal{T},0} \in \mathbb{R}^{1\times d}$, the fine-tuned $\boldsymbol{b}'_\mathcal{T}$ with LoRA is presented:

$$\boldsymbol{b}'_\mathcal{T} = \boldsymbol{b}_{\mathcal{T},0} + \Delta\boldsymbol{b}_\mathcal{T} = \boldsymbol{b}_{\mathcal{T},0} + \text{vec}(\underline{\boldsymbol{B}}\underline{\boldsymbol{A}}),$$

where $\boldsymbol{B} \in \mathbb{R}^{q\times r}$ and $\boldsymbol{A} \in \mathbb{R}^{r\times k}$ and $r < \min(q,k)$, as well as $q \times k = d$; $\boldsymbol{B}$ and $\boldsymbol{A}$ are the trainable parameters. Moreover, $\text{vec}(\boldsymbol{BA})$ means its vectorization, i.e., the operation $(\mathbf{B@A}).\text{flatten}()$ in PyTorch. The fine-tuned $\boldsymbol{b}'_\mathcal{T}$ with VeRA is presented as follow:

$$\boldsymbol{b}'_\mathcal{T} = \boldsymbol{b}_{\mathcal{T},0} + \Delta\boldsymbol{b}_\mathcal{T} = \boldsymbol{b}_{\mathcal{T},0} + \text{vec}(\underline{\Lambda_{\boldsymbol{i}}}\boldsymbol{B}\underline{\Lambda_{\boldsymbol{j}}}\boldsymbol{A}),$$

where $\boldsymbol{i} \in \mathbb{R}^{1\times q}$ and $\boldsymbol{j} \in \mathbb{R}^{1\times r}$ are trainable, and formally denoted by diagonal matrices $\Lambda_{\boldsymbol{i}}$ and $\Lambda_{\boldsymbol{j}}$. Moreover, to explore the extreme parameter efficiency, we exploit the variant of VeRA, namely VeRA_{1d}, directly applied to the $\boldsymbol{b}_{\mathcal{T},0}$, which is $\boldsymbol{b}'_\mathcal{T} = \boldsymbol{b}_{\mathcal{T},0} + \underline{\lambda}\cdot\boldsymbol{b}_{\mathcal{T},0}$. λ is a trainable scalar parameter.

$\boldsymbol{b}_\mathcal{T}$	+VeRA_{1d}
$\boldsymbol{b}_v$	**80.0**
$\boldsymbol{b}_q$	79.5
$\boldsymbol{b}_k$	79.5
ΔParams↓	1.3‰

Table 13: The downstream performance (%) of variants of VeRA_{1d} designed for BEFT on SST-2 with $\text{BERT}_{\text{BASE}}$, where $\boldsymbol{b}_v$ surpasses $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$.

The fine-tuned $\boldsymbol{b}'_\mathcal{T}$ with DoRA is presented as follow:

$$\boldsymbol{b}'_\mathcal{T} = \underline{m}\frac{\boldsymbol{v}+\Delta\boldsymbol{v}}{\|\boldsymbol{v}+\Delta\boldsymbol{v}\|_2} = \underline{m}\frac{\boldsymbol{b}_{\mathcal{T},0}+\text{vec}(\underline{\boldsymbol{BA}})}{\|\boldsymbol{b}_{\mathcal{T},0}+\text{vec}(\underline{\boldsymbol{BA}})\|_2},$$

where $m = \|\boldsymbol{b}_\mathcal{T}\|_2$ is the magnitude scalar parameter. $\boldsymbol{v} \in \mathbb{R}^{1\times d}$ is the directional vector and $\boldsymbol{v}/\|\boldsymbol{v}\|_2$ is a unit vector. $\Delta\boldsymbol{v}$ is the directional update learned by multiplying $\boldsymbol{B}$ and $\boldsymbol{A}$. m, $\boldsymbol{B}$ and $\boldsymbol{A}$ are the trainable parameters.

Models	(q,k,r)	Params Saving	$\boldsymbol{b}_v$	$\boldsymbol{b}_q$	$\boldsymbol{b}_k$
			+LoRA		
$\text{BERT}_{\text{BASE}}$	(24,32,8)	42%	**85.0**	82.6	76.9
$\text{RoBERTa}_{\text{BASE}}$	(32,24,8)	42%	**88.0**	86.0	79.5
$\text{BERT}_{\text{LARGE}}$	(32,32,8)	50%	**88.9**	77.1	69.3

Table 14: The downstream performance (%) of BEFT with LoRA on SST-2, where $\boldsymbol{b}_v$ surpasses $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$.

C.5 PiSSA and BA-LoRA for Bias Terms

We also performed extra experiments for our BEFT+PiSSA (Meng et al., 2024), and our BEFT+BA-LoRA (Chang et al., 2026) with $\text{RoBERTa}_{\text{BASE}}$ on SST-2 low data regime, which

highlights our key finding, i.e., $\boldsymbol{b}_v$ allows more effective fine-tuning than $\boldsymbol{b}_q$ and $\boldsymbol{b}_k$. In PiSSA and BA-LoRA, we reshape the bias terms from $\mathbb{R}^{1\times d}$ to $\mathbb{R}^{q\times k}$ (where $q \times k = d$) and then apply SVD to the bias terms and only fine-tune the first r rank components in two matrices ($\boldsymbol{A}$ and $\boldsymbol{B}$ format). Our results indicate that in BEFT+PiSSA (Meng et al., 2024), $\boldsymbol{b}_v$ achieves 15.7% higher accuracy compared to $\boldsymbol{b}_q$ and 22.4% higher accuracy than $\boldsymbol{b}_k$. In addition, in BEFT+BA-LoRA (Chang et al., 2026), $\boldsymbol{b}_v$ achieves 13.9% higher accuracy compared to $\boldsymbol{b}_q$ and 21.6% higher accuracy than $\boldsymbol{b}_k$.

C.6 Comparison with More Advanced PEFT Methods

OPT-1.3B	Params	SST-2 classification	ReCoRD multiple choice	SQuAD generation
$\boldsymbol{b}_v$	0.04‰	93.1	71.5	79.8
LoRA	1.2‰	93.6	71.0	79.7
VeRA	0.07‰	92.2	71.7	77.6
DoRA	1.3‰	93.8	71.8	80.9

Table 15: Downstream performance (%) of our BEFT compared to more advanced techniques such as VeRA (Kopiczko et al., 2024) and DoRA (Liu et al., 2024). Fine-tuning $\boldsymbol{b}_v$ requires 1.75× and 32.5× fewer parameters than VeRA and DoRA, respectively, while maintaining competitive downstream performance.

C.7 Extended to Multilingual and Commonsense Reasoning Datasets

For the multilingual dataset, we extract three different language sets ('en': English, 'zh': Chinese, 'fr': French) from PAWS-X data, with the XLM-RoBERTa model. In low-data regime, we consider 3333 training samples for each language set and a total of 1998 test samples from different language sets. Our results indicate that for PAWS-X datasets, fine-tuning $\boldsymbol{b}_v$ achieves over 9.1% higher accuracy, compared to $\boldsymbol{b}_q$ or $\boldsymbol{b}_k$. In addition, we have performed initial experiments on CommonsenseQA, which is a question answering dataset for commonsense reasoning, and observed that fine-tuning $\boldsymbol{b}_v$ achieves over 17.6% higher accuracy, compared to $\boldsymbol{b}_q$ or $\boldsymbol{b}_k$.

C.8 Adding Bias Terms to More Bias-Free LLMs

GPT-J-6B (Wang and Komatsuzaki, 2021) does not include bias terms in its attention module, and DeepSeek-Coder-Base-1.3B (Guo et al., 2024) does not include bias terms in the whole model.

Bias-Free LLMs	Adding $\boldsymbol{b}_v$	Adding $\boldsymbol{b}_q$	Adding $\boldsymbol{b}_k$
DeepSeek-Coder-Base-1.3B	**76.9**	67.4	60.3
GPT-J-6B	**92.8**	88.3	63.8

Table 16: Downstream performance (%) on SST-2 dataset by adding bias terms into bias-free LLMs, showing that for bias-free LLMs in the low-data regime, *directly adding* $\boldsymbol{b}_v$ and fine-tuning $\boldsymbol{b}_v$ is sufficient without requiring any post-hoc evaluation.

D Experimental Details

We first define different training data regimes on the GLUE benchmark in Table 17. Then we present the hyperparameters for $\text{BERT}_{\text{BASE}}$ on the small dataset from the GLUE benchmark in Table 18 and demonstrate hyperparameters for $\text{BERT}_{\text{BASE}}$ on the large dataset from the GLUE benchmark in Table 19. Next, we show the hyperparameters for $\text{BERT}_{\text{BASE}}$ on WiC and CB datasets from the SuperGLUE benchmark on the low-data regime in Table 20. Meanwhile, we show the hyperparameters for $\text{RoBERTa}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ on SST-2 and RTE datasets from GLUE benchmark, and WiC and CB from SuperGLUE benchmark on the low-data regime in Table 21. Finally, we present the hyperparameters for OPT-1.3B and OPT-6.7B in Table 22.

	Low-Data	Medium-Data	High-Data
SST-2	1000	5000	11000
RTE	300	900	1200
QQP	1000	5000	9000
QNLI	1000	5000	9000
MNLI_m	1000	5000	9000
MNLI_{mm}	1000	5000	9000
CoLA	1710	3420	4275
MRPC	367	1191	1835
STS-B	287	1150	2300

Table 17: The definition of different training data regimes on the GLUE benchmark.

	Train Size	Val Size	lr	Batch Size	Train Epoch $\boldsymbol{\theta}$	Train Epoch $\boldsymbol{b}_{\text{all}}$	Train Epoch $\boldsymbol{b}_{\mathcal{T}}$	Metrics
SST-2	67k	872	4e-4	16	16	16	16	Accuracy
RTE	2.5k	277	1e-3	16	30	30	30	Accuracy
CoLA	8.5k	1k	7e-4	16	16	16	16	MCC
MRPC	3.7k	408	7e-4	16	16	16	60	F1
STS-B	5.7k	1.5k	1e-4	16	16	30	60	SCC

Table 18: Hyperparameters for $\text{BERT}_{\text{BASE}}$ on small dataset from GLUE benchmark (MCC denotes Matthews Correlation Coefficient; SCC denotes Spearman Correlation Coefficient).

	Train Size	Val Size	lr	Batch Size	Train Epoch $\boldsymbol{b}_{\text{all}}$	Train Epoch $\boldsymbol{b}_{\mathcal{T}}$	Metrics
QQP	364k	40k	4e-4	16	16	30	F1
QNLI	105k	5.5k	1e-4	16	16	30	Accuracy
MNLI_m	393k	9.8k	1e-4	16	16	30	MA
MNLI_{mm}	393k	9.8k	1e-4	16	16	30	MMA

Table 19: Hyperparameters for $\text{BERT}_{\text{BASE}}$ on large dataset from GLUE benchmark (MA denotes Matched Accuracy; MMA denotes Mismatched Accuracy).

	Low-Data Regime	Val Size	lr	Batch Size	Train Epoch $\boldsymbol{b}_{all}$	Train Epoch $\boldsymbol{b}_{\mathcal{T}}$	Metrics
WiC	1000	638	1e-3	16	16	16	Accuracy
CB	250	56	1e-3	16	16	60	F1 (macro)

Table 20: Hyperparameters for $\text{BERT}_{\text{BASE}}$ on WiC and CB datasets from SuperGLUE benchmark on low-data regime.

	Low-Data Regime	Val Size	lr	Batch Size	Train Epoch $\boldsymbol{b}_{\mathcal{T}}$	Metrics
SST-2	1k	872	4e-4	16	30	Accuracy
RTE	300	277	1e-3	16	30	Accuracy
WiC	1000	638	1e-3	16	30	Accuracy
CB	250	56	1e-3	16	60	F1 (macro)

Table 21: Hyperparameters for $\text{RoBERTa}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ on SST-2 and RTE datasets from GLUE benchmark, and WiC and CB from SuperGLUE benchmark on low-data regime.

	Low-Data Regime	Val Size	Model Adaptation	Learning Rate (lr)	Metrics
SST-2	1k	872	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA*/Prefix	1e-3	Accuracy
RTE	1k	277	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA*/Prefix	1e-3	Accuracy
BoolQ	1k	1k	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA*/Prefix	1e-3	Accuracy
MultiRC	1k	1k	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA*/Prefix	1e-3	Accuracy
WSC	554	104	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA/Prefix	1e-2	Accuracy
CB	250	56	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA*/Prefix	1e-2	Accuracy
WiC	1k	638	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA*/Prefix	1e-3	Accuracy
COPA	400	100	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA/Prefix	1e-4	Accuracy
ReCoRD	1k	1k	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA/Prefix	1e-4	Accuracy
SQuAD	1k	1k	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA/Prefix	1e-3	F1
DROP	1k	1k	$\boldsymbol{b}_v/\boldsymbol{b}_q/\boldsymbol{b}_k$/LoRA/Prefix	1e-3	F1

Table 22: Hyperparameters for OPT-1.3B and OPT-6.7B (default epoch is 5; default batch size is 8; default max length the model can take is 2048, while 1536 for OPT-6.7B on DROP dataset, and small batch size with gradient accumulation is adopted for OPT-6.7B (Marek et al., 2025) due to the Out-of-Memory (OOM) problem). For LoRA*, the learning rate is 1e-4.