Enhancing Parameter-efficient Fine-tuning with Simple Calibration based on Stable Rank

Peiyu Liu[1,2], **Ze-Feng Gao**[1,3], **Xiao Zhang**[1,2], **Wayne Xin Zhao**[1,2*], **Ji-Rong Wen**[1,2,4]
[1] Gaoling School of Artificial Intelligence, Renmin University of China
[2] Beijing Key Laboratory of Big Data Management and Analysis Methods
[3] Department of Physics, [4] School of Information, Renmin University of China
liupeiyustu@163.com, zfgao,zhangx89,jrwen@ruc.edu.cn, batmanfly@gmail.com

Abstract

Lightweight fine-tuning is widely used as an important technique for efficiently adapting pre-trained language models (PLM) to downstream tasks. Despite the reduction in trainable parameters, existing lightweight fine-tuning methods are found to be effective in low-resource settings but often fail in high-resource settings, leading to unreliable outcomes. This limitation can be attributed to inflexible strategies: they identify the parameters of the model to be trained before fine-tuning and remain unchanged without taking into account the inherent variance of generalization ability in model components (*i.e.*, feed-forward, attention layers) and potential changes during the fine-tuning process. In this paper, we introduce a simple but effective calibration for lightweight fine-tuning PLMs based on the matrix's stable rank according to both model components and the training process. We proposed both theoretical analyses and experimental verification for the proposed calibration strategy. Considering efficiency, we further propose time-aware and structure-aware strategies to determine the most crucial time to commence the fine-tuning procedure and selectively apply parameter matrices for lightweight fine-tuning, respectively. Extensive experiments demonstrate the superiority of our proposed fine-tuning approach (average improvement 3.1 for GLUE score compared to lightweight fine-tuning method).

Keywords: pre-trained language models, lightweight fine-tuning, stable rank

1. Introduction

With recent development in pre-trained language models (PLMs), the approach of "*pre-training and fine-tuning*" has become a de-facto paradigm in the field of natural language processing (NLP) field (Devlin et al., 2018; Raffel et al., 2020; Zhao et al., 2023). Despite the effectiveness, the increasingly large model size makes it expensive to keep separate copies of all fine-tuned parameters, *i.e., full fine-tuning (FFT)*, for various downstream tasks. To reduce the heavy overload, recent studies have found that it is feasible to conduct *lightweight fine-tuning (LFT)*, also known as parameter-efficient fine-tuning, which only update a small subset and leaving the rest intact (Aghajanyan et al., 2021; Liu et al., 2021a,b; Li and Liang, 2021; Liu et al., 2023a) for fine-tuning.

Compared with FFT, LFT is significantly more efficient by only updating part of model parameters. In addition, LFT has the potential to improve the generalization abilities of models, since FFT tends to lead to the *overfitting* (Jiang et al., 2020) of PLMs on downstream tasks with limited training data (Devlin et al., 2018; Lee et al., 2020). In the exist literature, various LFT methods have been proposed, which mainly focus on different strategies to select the parameters for fine-tuning, including inserting external structures (Houlsby et al., 2019; Li and Liang, 2021), measuring the entry importance (Ding et al., 2022) and decomposing the parameter matrices (Liu et al., 2021a).

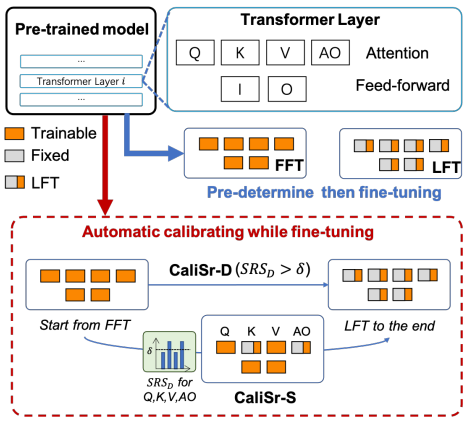


Figure 1: Overall comparison of FFT, LFT and CaliSr approach in fine-tuning PLMs. SRS_D denotes the metric to determine when and where to start lightweight fine-tuning.

Despite the effectiveness, there are two major shortcomings for existing fine-tuning approaches. First, these LFT methods adopt a relatively rigid optimization process, without considering task difficulty or model capacity during fine-tuning. Therefore, it can be often observed that LFT methods have slow convergence rate and under-perform FFT methods, especially in high-resource set-

* Corresponding author.

LREC-COLING 2024, pages 6024–6035
20-25 May, 2024.

tings (Chen et al., 2022; He et al., 2021). This constraint renders the LFT method susceptible to yielding unreliable conclusions, primarily because it is hard to know whether the existing data volume is sufficient for the current task. Second, there still lacks a deep understanding of why LFT can improve the model generalization abilities, though it has been well-known as an advantage over FFT. Therefore, it is imperative to develop a theoretically supported LFT approach that allows for more flexible control or careful selection over the parameters to be optimized during the fine-tuning process.

To address the above issues, we focus on developing a more principled LFT approach, with both *flexible adaptation* and *theoretical guidance* for pre-trained language models (PLMs). The key idea is to combine the benefits of both FFT and LFT: FFT can speed up the convergence, especially at the beginning stage of training or on complex tasks, whereas LFT can efficiently adapt to different tasks starting from a good model checkpoint. In this way, our fine-tuning approach always starts with FFT, and further develops a theoretical criterion to decide when to switch to the LFT, based on different components in PLMs. Such a way can achieve a good trade-off between both effectiveness and efficiency in fine-tuning PLMs.

To this end, in this paper, we propose a theoretically-informed **Cali**bration approach for LFT based on matrix **S**table **R**ank, called **CaliSr**. Specially, we incorporate the *matrix stable rank* (Sanyal et al., 2020) as the measure to determine the start time of LFT. This metric plays a vital role in identifying the precise conditions under which a more stringent upper bound can be established in relation to generalization error (Bartlett et al., 2017; Neyshabur et al., 2018), which are easy to compute during fine-tuning. Further, we devise two stable rank instructed approaches to adaptively starting the LFT process for updating the corresponding parameters from different model components. In contrast to prior methods that rely on predetermined fine-tuning parameters (Li and Liang, 2021; Houlsby et al., 2019; Xu et al., 2021), our approach can dynamically select the most appropriate parameter matrices to update during the fine-tuning process. In addition, we also provide the theoretical analysis why LFT can achieve a better performance than FFT, based on stable rank.

To the best of our knowledge, this is the first time that matrix stable rank has been applied to improve PLM fine-tuning, which is well suited to explain why current lightweight fine-tuning is effective. We construct extensive experiments to evaluate the effectiveness of the proposed CaliSr approach for BERT on GLUE benchmark, compared with Adapters (Houlsby et al., 2019), Prefix-tuning (Li and Liang, 2021), ChildTuning (Xu et al., 2021), MPOP (Liu et al., 2021a), BitFit (Zaken et al., 2021), Diff-Prune (Guo et al., 2021), LoRA (Hu et al., 2022), and Adapters (Houlsby et al., 2019). It is demonstrated that the proposed CaliSr approach can achieve better accuracy for downstream tasks (average improvement of 3.1 for GLUE score compared with LoRA, Adapters, Prefix-tuning, MPOP and BitFit). All the experimental codes are available on `https://github.com/RUCAIBox/CaliSr`.

2. Related Work

The existing LFT approaches can be generally divided into modular methods, sparse fine-tuning, and matrix decomposition-based methods.

Modular methods. Modular methods added new parameters in the form of extra modules and fine-tuned while keeping all the pre-trained parameters fixed. Adapters formulated alternative trainable parameters as extra intermediate layers (Houlsby et al., 2019; Rebuffi et al., 2017) while prefix-tuning (Li and Liang, 2021) used prefix. LoRA (Hu et al., 2022) leverages both the low-rank approximation method and the Adapter technique to significantly decrease the number of fine-tunable parameters (Liu et al., 2023b). Liu et al. (2021b) used prompts also make the tuning process more efficient. In contrast, our approach intended to improve model performance by the stable rank metric to find the more efficient and robust tuning approach for PLMs.

Sparse fine-tuning. Unlike modular methods which modify the model architecture, sparse fine-tuning selects important values of parameter matrices by applying the sparse binary mask. As a conventional technique, the binary mask is employed to modify pre-trained weights (Zhao et al., 2020), and its learning can be achieved through fine-tuning or by evaluating the Fisher information for each individual value (Xu et al., 2021). Besides, the diff-pruning method applies a binary mask to the difference vector learned by downstream tasks (Guo et al., 2021), while the BitFit approach only modifies bias items instead of weight matrices (Zaken et al., 2021). Furthermore, the MoEfication method was proposed due to the sparse activation in PLMs (Zhang et al., 2021). Fortunately, our approach could combine these methods to improve model performance even further.

Matrix decomposition-based methods. Matrix decomposition is a common technique in machine learning. Low-rank approximation by decomposition, such as SVD (Henry and Hofrichter, 1992)

can reduce redundancy for those deep neural networks with low-rank properties (Gao et al., 2020, 2022, 2023). Recently, Aghajanyan et al. (2021) demonstrated that the trained over-parametrized models have a low inherent dimension. Inspired by this, LoRA optimized decomposed low-rank matrices while keeping the pre-trained weights frozen to make computation efficient (Hu et al., 2022). Moreover, MPOP (Liu et al., 2021a) achieved efficient fine-tuning by only modifying part of decomposed tensors. In all these approaches it is possible to achieve approximate full fine-tuning with a very small training cost. Our approach was based on these works in that we combined the low-rank approximation and the stable rank to construct a more efficient fine-tuning approach.

3. Approach

In this section, we describe our proposed *CaliSr* based on matrix stable rank for improving the fine-tuning performance of PLMs. We first give an overview of our approach, then introduce the details of CaliSr strategy and theoretical analysis.

3.1. Background

We focus on the topic of model fine-tuning under the setting of pre-trained language models (PLMs), which can be approached through two main methods, namely *full fine-tuning (FFT)* and *lightweight fine-tuning (LFT)*. Typically, the LFT process starts from the pre-trained model checkpoint and continues to update the selected parameters until reaching the stop condition. Compared with FFT, LFT saves the computational cost by reducing the number of trainable parameters, meanwhile achieving a decent performance. Despite the effectiveness, recent studies (Chen et al., 2022; He et al., 2021) demonstrate that LFT sometimes suffers from performance degradation or slow convergence. In addition, it also requires to pre-determine the model components (*e.g.,* only query and value in attention for LoRA (Hu et al., 2022)) to update.

3.2. Stable Rank for Model Generalization

In this part, we first introduce a matrix based measure, *stable rank*, which will be used to guide the fine-tuning process in our approach.

Formally, stable rank (Sanyal et al., 2020) is defined as the squared ratio between the Frobenius norm and the spectral norm, formulated as follows:

$$\text{srank}(\mathbf{W}) = \frac{\|\mathbf{W}\|_F^2}{\|\mathbf{W}\|_2^2} = \frac{\sum_{j=1}^{k} \sigma_j^2(\mathbf{W})}{\sigma_1^2(\mathbf{W})}, \quad (1)$$

where k is the rank of an arbitrary matrix $\mathbf{W}$, $\|\mathbf{W}\|_2$ denotes the spectral norm of $\mathbf{W}$ and $\sigma_i(\mathbf{W})$ denotes the i-th singular value of the matrix $\mathbf{W}$.

Our approach is inspired by the connection between stable rank and the model generalization bound for a d-layer neural network (Sanyal et al., 2020): $\mathcal{O}\left(\sqrt{\prod_i^d \|\mathbf{W}_i\|_2^2 \sum_{i=1}^d \text{srank}(\mathbf{W}_i)}\right)$. In this bound, we can see that stable rank is a key indicator for the generalization capacity of a neural network since it can directly affect the generalization behaviour. Sanyal et al. (2020) further demonstrates that decreasing stable rank leads to improved generalization in practice.

Note that we are not concerned with the actual generalization bound, but instead we employ stable rank as an indicator to monitor the fine-tuning process of a parameter matrix. Intuitively, the smaller the stable rank is, the training of a parameter matrix is more sufficient (approaching the convergence).

3.3. Stable Rank Instructed Fine-tuning

In this part, we explore the use of stable rank as an instruction measure to guide the selection of fine-tuning parameters.

Underlying Principle. In our approach, we consider the fine-tuning process is a mixture of FFT and LFT. For each parameter matrix, the training always starts by FFT, and it switches to LFT when the training becomes relatively sufficient. To determine when to make the switch, we introduce the *Stable Rank Score (SRS)* as a means of determining the starting point for LFT in PLMs. The underlying principle is that a parameter matrix that is ready for LFT should have a lower stable rank score than that obtained after FFT, because FFT becomes more likely to be overfitted as the training process continues (thus leading to a more poor generalization than LFT). We refer to the monitor process that switches between FFT and LFT as *calibration*, and propose two calibration approaches, namely dynamic calibration and structure-aware calibration, which will be detailed next.

Dynamic Calibration (CaliSr-D). The proposed dynamic calibration, *CaliSr-D*, uses the stable rank metric as an indicator for determining the optimal starting point for LFT in PLMs. A straightforward approach is compute stable rank metric at each training step until the target condition is met. However, this method may not be stable due to the randomness of optimization process, which is also computationally costly. To address these issues, we introduce a new strategy by calculating the indicator multiple times over the calibration process and counting the count that times the condition is satisfied. If the count exceeds a specific threshold,

LFT would be initiated. The count is calculated as:

$$SRS_D = \frac{\sum_{i=1}^{n} \mathbb{1}\left(SR^{(i)}(\mathbf{W}_{LFT}) < SR^{(i)}(\mathbf{W}_{FFT})\right)}{n}, \tag{2}$$

where n denotes the times of SRS calculation in the calibration process ($n \in \{3, 5\}$ for our experiments). In this way, we can reduce the random risk during the optimization process and derive a more reliable stable rank score.

Structure-aware Calibration (CaliSr-S). Our proposed structure-aware calibration, *CaliSr-S*, aimed to calculate the stable rank separately for different structures within the PLMs. In general, the Transformer model contains two basic components, namely attention modules (*i.e.,* $\mathbf{W}_Q$,$\mathbf{W}_K$,$\mathbf{W}_V$ and $\mathbf{W}_{AO}$) and feed-forward modules (*i.e.,* $\mathbf{W}_I$ and $\mathbf{W}_O$). In our approach, we aggregate the stable rank of the weights across all layers to obtain an overall score for the parameter matrices in each component. This approach provides a more fine-grained control over the switch between FFT and LFT for different components, thus likely to achieve a better fine-tuning performance. Unlike prior studies (Hu et al., 2022; Liu et al., 2021a) that predetermine the fine-tuning parameters and use a fixed LFT fine-tuning way, our approach makes it feasible to adaptively set the start time of LFT process for different components in PLMs.

Training. In this part, we provide a complete description of our calibration process. Without loss of generality, we reparameterize the model parameters as $\mathbf{W}^{(t)} = \mathbf{W}^{(t)}_{core} + \mathbf{W}^{(t)}_{extra}$. In this decomposition, $\mathbf{W}^{(t)}$ denotes the total parameters at the t-th iteration, and during LFT, only $\mathbf{W}^{(t)}_{core}$ is trained, whereas during FFT, both $\mathbf{W}^{(t)}_{core}$ and $\mathbf{W}^{(t)}_{extra}$ are trained. As discussed before, our approach first performs parameter update via FFT. Subsequently, the update process can be divided into two stages. In the first stage, we only update the weights $\mathbf{W}^{(t)}_{core}$, which can be approximated as an LFT update strategy; in the second stage, we update the weights $\mathbf{W}^{(t)}_{extra}$, which can be approximated as an FFT update strategy. By employing this approach, we can effectively control how to make the switch between LFT and FFT. Following the completion of the calibration process, the model will continue employing the LFT techniques. Overall, our proposed approach combines the benefits of both FFT and LFT to achieve improved results on downstream tasks. The complete fine-tuning procedure is shown in Algorithm 1.

Theoretical Analysis. We now apply the generalization bound based on stable rank, as discussed earlier in Section 3.2, to clarify the reasoning behind using the SRS_D metric.

Algorithm 1 The proposed CaliSr approach.

Input: n: interval steps for calculating the stable rank;δ: threshold for start lightweight fine-tuning; $steps$: total steps in each epoch.

Initialize timestamp $t \leftarrow 0$.
Training with FFT
while not start LFT **do**
  Fine-tune $model$ with all parameters
  $t \leftarrow t + 1$
  if $t\%n = 0$ **then**
    Update $\mathbf{W}^{(t)}_{core}$
    Calculate stable rank $SR(\mathbf{W}_{LFT})$
    Update $\mathbf{W}^{(t)}_{extra}$
    Calculate stable rank $SR(\mathbf{W}_{FFT})$
  end if
  if $t\%steps = 0$ **then**
    Evaluate SRS
    if $SRS > \delta$ **then**
      Start lightweight fine-tuning
    end if
  end if
end while
Training $model$ with lightweight fine-tuning
return $model$

As previously mentioned, Sanyal et al. (2020) has defined a generalization bound as follows: $\mathcal{O}\left(\sqrt{\prod_i^d \|\mathbf{W}_i\|_2^2 \sum_{i=1}^d \text{srank}(\mathbf{W}_i)}\right)$. This bound depends on both the upper bound of the Lipschitz constant, denoted as $\prod_i^d |\mathbf{W}_i|_2^2$, and the stable rank. Furthermore, Sanyal et al. (2020) has empirically demonstrated that as the stable rank decreases, the Lipschitz constant also decreases. Hence, these findings suggest that it is reasonable to assess model generalization primarily by monitoring the stable rank of the weight matrices. This analysis emphasizes that by evaluating SRS_D, we can make a selective transition between FFT and LFT techniques, ensuring a consistent improvement in model generalization.

3.4. Investigating the Mechanism of Lightweight Fine-tuning

Prior study (Chen et al., 2022) has shown that lightweight fine-tuning (LFT) can achieve competitive results in low-resource settings, but its underlying mechanisms remain less explored. Specifically, the working mechanism of LFT, which enables training large models with minimal computational cost, has not been fully understood. In this section, we aim to analyze the effectiveness of LFT from the perspective of generalizability in relation to the stable rank of model weights.

Formally, we first introduce the notation $\Delta\mathbf{W}$, which represents the difference between matrices obtained from fine-tuning and pre-trained weights, *i.e.,* $\Delta\mathbf{W} = \mathbf{W}_{FT} - \mathbf{W}$. We then propose a theo-

rem that identifies the conditions under which fine-tuning achieves a lower stable rank. We denote $\{\sigma_i\}$ and $\{a_i\}$ as eigenvalues of $\mathbf{W}_{FT}$ and $\mathbf{\Delta W}$ by SVD, respectively. Then, the srank$(\mathbf{W}_{FT}) <$ srank$(\mathbf{W})$ holds if

$$\frac{a_1 + \sigma_1}{\sqrt{\sum_{i=2}^{r}(\sigma_i + a_i)^2}} > \frac{\sigma_1}{\sqrt{\sum_{i=2}^{r}\sigma_i^2}}, \tag{3}$$

where r denotes the rank of pre-trained weights. Equation (3) suggests that the key to further improving model generalization by fine-tuning is to adjust the distribution of eigenvalues rather than the magnitude of the eigenvalues. If the weight difference in the LFT process exhibits a bigger ratio compared to that of the FFT, LFT would have a lower stable rank, indicating a reduced generalization error. A comprehensive proof and further details will be found in the Appendix. Specially, we assess the ratio described in Equation (3) for BERT across various downstream tasks, as detailed in Section 4.3. The results consistently validate that LFT exhibits a lower stable rank in practice.

Additionally, our analysis also reveals a key insight into the efficacy of LFT in updating large models: it selectively modifies non-dominant eigenvalues to lower the stable rank. As a comparison, FFT randomly alters the amplitudes of all eigenvalues, resulting in a similar stable rank but incurring higher computational costs. This targeted updating strategy contributes to the efficient nature of LFT, as it achieves comparable results with reduced computational costs.

4. Experiments

In this section, we present the evaluation of the CaliSr approach on various datasets, providing results and detailed analysis.

4.1. Experimental Setup

Datasets. We conduct our evaluation on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019). To assess the performance of our approach, we report the Matthews correlation coefficient for CoLA, Separnman correlation for STS-B, F1 score for MRPC/QQP, and accuracy for the remaining tasks. The test sets for these tasks are provided by the GLUE evaluation server. We report the mean values of five runs using different random seeds for validation results.

Baseline Methods. We apply CaliSr on LFT approaches that modify the original weights (MPOP, LoRA, and ChildTuning) rather than those that change model architecture (Adapters). This choice avoids the need for frequent modifications to the model architecture during the calibration process. We consider the following baseline methods:

- MPOP (Liu et al., 2021a): It applies MPO to model weights and selects a small part of produced local tensors (called "auxiliary tensors") and fine-tuned on downstream tasks.
- LoRA (Hu et al., 2022): It injects trainable rank decomposition matrices into each layer of the transformer architecture while freezing the pre-trained weights, greatly reducing the number of trainable parameters for downstream tasks.
- ChildTuning (Xu et al., 2021): It adopts the Fisher information estimation to select the important parameters at the beginning of the fine-tuning.

In addition, we also implement other LFT approaches such as Adapters (Houlsby et al., 2019), Prefix-tuning (Li and Liang, 2021) and BitFit (Zaken et al., 2021) for comparison. For a fair comparison, we compare the performance of these methods based on the same checkpoint (*i.e.,* $\text{BERT}_{\text{base}}$, $\text{BERT}_{\text{large}}$ and $\text{RoBERTa}_{\text{large}}$.

4.2. Main Experimental Results

In this part, we report and analyze the experimental results on $\text{BERT}_{\text{large}}$.

Overall Performance. In our experiments on the GLUE benchmark, we compared the performance of our CaliSr approach with both current LFT methods and FFT. The results, presented in Table 1, demonstrate that our approach consistently outperforms these methods in terms of validation scores. On average, our approach achieves a 2.1 improvement over MPOP and a 1.1 improvement over full fine-tuning. Notably, the combination of MPOP and CaliSr-S performs the best among all LFT methods in terms of the average GLUE score. Furthermore, for the test sets, the "MPOP+CaliSr-S" approach even achieves a 0.1 higher score compared to the FFT method.

Effectiveness on Different LFT methods. We demonstrate the compatibility of the CaliSr approach with other fine-tuning methods. Specifically, we integrate the CaliSr approach as a plugin to enhance existing methods, namely MPOP, ChildTuning and LoRA, and present the results in Table 2. Our results indicate that the CaliSr-S variant outperforms the vanilla LFT method on all three datasets, with the greatest improvement observed on the RTE dataset (*i.e.,* 5.2 higher score compared to LoRA). However, a decline in performance is sometimes observed for the CaliSr-D, suggesting that structural differences may have a significant impact in this scenario.

Exp	SST-2	MRPC	CoLA	QNLI	RTE	MNLI	QQP	STS-B	Avg.
Development Set Result									
Adapters (Houlsby et al., 2019)	89.9	86.9	51.2	91.1	68.2	85.1	85.3	88.7	80.8
LoRA (Hu et al., 2022)	93.6	89.8	49.9	89.0	69.9	83.0	83.0	86.5	80.6
Prefix-tuning (Li and Liang, 2021)	92.9	85.6	50.8	90.2	68.6	82.8	84.0	89.0	80.5
BitFit (Zaken et al., 2021)	93.2	91.7	63.6	91.4	73.2	84.4	85.4	**90.3**	84.2
MPOP (Liu et al., 2021a)	93.5	90.0	57.9	90.7	73.3	84.7	85.5	89.4	83.1
Full Fine-Tuning	93.4	90.7	62.2	91.7	71.9	85.5	87.5	90.0	84.1
MPOP+CaliSr-D	93.6	92.3	62.7	92.2	74.3	85.6	87.6	90.1	84.8
MPOP+CaliSr-S	**94.3**	**92.3**	**62.7**	**92.2**	**76.2**	**85.8**	**87.6**	90.1	**85.2**
Test Set Result									
Full Fine-Tuning	94.1	**88.9**	59.6	91.1	**71.2**	**86.7**	71.7	**86.6**	81.2
MPOP+CaliSr-D	94.1	88.4	59.9	92.3	70.6	84.8	71.9	86.3	81.0
MPOP+CaliSr-S	**94.9**	88.4	**59.9**	**92.3**	70.7	85.9	**71.9**	86.3	**81.3**

Table 1: Performance comparison using $\text{BERT}_{\text{large}}$ on GLUE benchmark (in percent). "MPOP+CaliSr-D" represents the use of the CaliSr approach with the time-aware variant, whilst "MPOP+CaliSr-S " signifies the use of the structure-aware variant. Bold fonts indicate the best results in each block. We report the mean values of five runs using different random seeds for all the results.

Exp	SST-2	MRPC	RTE
MPOP (Liu et al., 2021a)	93.5	90.0	73.3
+CaliSr-D	93.8	91.5	70.0
+CaliSr-S	**94.3**	**92.3**	**76.2**
ChildTuning (Xu et al., 2021)	93.0	90.3	73.3
+CaliSr-D	93.0	90.5	71.9
+CaliSr-S	**93.3**	**91.3**	**74.0**
LoRA (Hu et al., 2022)	93.6	89.8	69.9
+CaliSr-D	93.6	90.1	71.1
+CaliSr-S	**93.8**	**91.9**	**75.1**

Table 2: Evaluation with different lightweight fine-tuning methods.

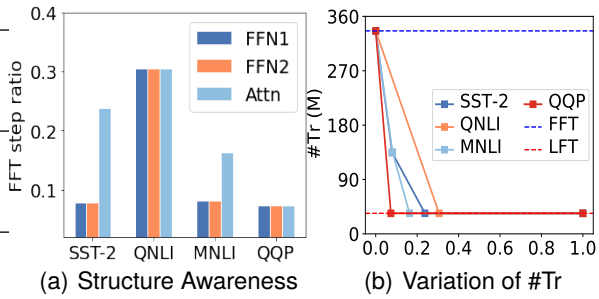


Figure 2: (a). Structure awareness *w.r.t.* the training process. (b). Variations of trainable parameters *w.r.t.* the training process.

Effectiveness on Different PLM. To illustrate the effectiveness of the different base models we based on, we also evaluate on $\text{BERT}_{\text{base}}$ and $\text{RoBERTa}_{\text{base}}$. The results on the GLUE benchmark are presented in Table 3, where we compare the performance of CaliSr with existing lightweight fine-tuning schemes. The results clearly demonstrate that the CaliSr approach consistently outperforms other methods in terms of the average score, particularly for CaliSr-S. This indicates the effectiveness and generality of our approach across different base models. The superior performance of CaliSr-S further supports the idea that the calibration is tailored to the specific characteristics of the model in terms of both time and structure.

Effectiveness of Structure Awareness. Here we analyze the performance of different tasks and evaluate the effectiveness of our CaliSr-S strategy in terms of structure awareness. Specifically, Figure 2(a) shows that the optimal time to begin LFT varies across different structures (*i.e.,* the attention module differs from others in SST-2 and MNLI), which highlights the existence of inherent structural differences. Additionally, our method also demonstrates adaptability by adjusting the starting time for different tasks. Based on the dynamic selection of CaliSr, Figure 2(b) shows the variation of trainable parameters with the FFT step ratio, which indicates that CaliSr lies between FFT and LFT and will soon reach a level similar to that of LFT (*e.g.,* MPOP only trains 10% of total parameters).

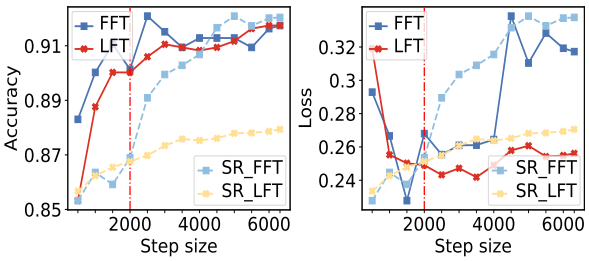
4.3. Further Analysis

Furthermore, we will conduct a more detailed analysis to further investigate the effectiveness of our proposed approach.

Analysis of Connection Between Stable Rank and Model Generalizability. To further justify our use of stable rank as an indicator of model generalization, we conducted experiments to analyze the dynamic evolution of stable rank, loss, and accuracy under LFT and FFT. Figure 3 provides visual-

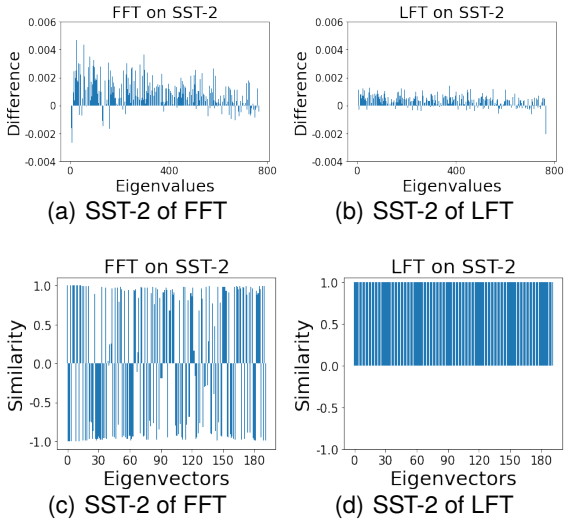
Experiments	SST-2	MRPC	CoLA	QNLI	RTE	MNLI	QQP	STS-B	Avg.
BERT$_{base}$									
Adapters (Houlsby et al., 2019)	89.9	87.8	54.4	90.8	60.2	83.5	85.8	88.2	80.1
LoRA (Hu et al., 2022)	86.7	82.1	40.9	76.0	60.3	56.7	68.9	81.5	69.1
Prefix-tuning (Li and Liang, 2021)	92.4	87.7	47.0	90.5	58.8	82.9	84.0	88.4	79.0
BitFit (Zaken et al., 2021)	87.2	81.2	55.1	88.9	47.3	81.2	85.7	**88.9**	76.9
MPOP (Liu et al., 2021a)	91.2	86.1	58.4	90.1	59.9	81.4	83.7	88.1	79.9
MPOP+CaliSr-D	92.4	89.3	59.0	91.3	60.8	81.6	87.0	88.1	81.1
MPOP+CaliSr-S	**92.4**	**89.3**	**59.0**	**91.3**	**60.8**	**84.6**	**87.0**	88.1	**81.6**
RoBERTa$_{base}$									
Adapters (Houlsby et al., 2019)	93.3	91.1	54.6	**92.3**	73.4	86.9	90.3	90.2	84.0
LoRA (Hu et al., 2022)	**94.2**	91.0	53.7	91.0	69.7	87.0	**90.5**	**90.5**	83.5
Prefix-tuning (Li and Liang, 2021)	**94.2**	89.4	53.4	**92.3**	57.4	86.3	88.9	89.6	81.4
BitFit (Zaken et al., 2021)	93.9	91.1	55.6	91.6	70.7	85.5	89.3	90.4	83.5
MPOP (Liu et al., 2021a)	92.2	90.2	53.6	91.3	52.7	86.4	89.4	90.3	80.8
MPOP+CaliSr-D	93.9	**92.1**	55.0	91.6	68.2	**87.1**	86.9	90.1	83.1
MPOP+CaliSr-S	93.9	**92.1**	**59.4**	92.2	**75.1**	**87.1**	87.2	90.6	**84.7**

Table 3: Performance comparison using BERT$_{\text{base}}$ and RoBERTa$_{\text{base}}$ on GLUE benchmark (in percent). The best performance in each group is highlighted in bold. For all the results, we report the mean values of five runs using different random seeds.



(a) Accuracy and Stable Rank (b) Loss and Stable Rank

Figure 3: Relation between the model stable rank and performance of SST-2 task. The red dashed line is the best option suggested by the total stable rank of the model.



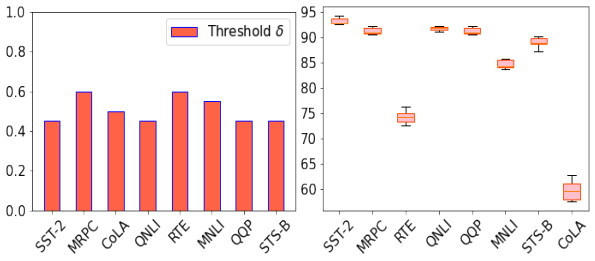
(a) SST-2 of FFT (b) SST-2 of LFT

(c) SST-2 of FFT (d) SST-2 of LFT

Figure 4: (a,b): Comparison of the difference of eigenvalues between fine-tuned weights and pre-trained weights. (c,d): Comparison of cosine similarity of eigenvectors between fine-tuned weights and pre-trained weights.

izations of these metrics. First, we assessed the generalization behavior of the model by analyzing the loss and accuracy trends. Our findings indicate that there is a notable difference in the loss curve between FFT and LFT. After 2000 steps, FFT exhibits a significant increase in loss and consistently performs worse than LFT in terms of loss. This suggests that LFT exhibits better generalization capabilities after 2000 steps. Similarly, we observed consistent variations in the stable rank curve. The stable rank of FFT is significantly higher than that of LFT after 2000 steps. This observation aligns with the inferior performance of FFT in terms of loss and implies a potential correlation between stable rank and model generalization. These experimental findings provide compelling empirical evidence supporting the utility of stable rank as a meaningful indicator of model generalization.

Empirical Results on Mechanism of Lightweight Fine-tuning. To illustrate the advantages of the LFT over the FFT, we calculate the ratio of eigenvalues, $\frac{a_1+\sigma_1}{\sqrt{\sum_{i=2}^{r}(\sigma_i+a_i)^2}}$ as discussed in Eq. 3 of Section 3.4, and visualize the results in Figure 4(a) and Figure 4(b). It can be observed that the LFT method increases the top eigenvalue of the pre-training weights, while the FFT method decreases these eigenvalues, indicating LFT has a bigger ratio than FFT. We also notice that the overall magnitude modification of the eigenvalues by the LFT is



(a) The best threshold δ (b) Sensitivity of threshold

Figure 5: Comparison of different threshold δ in different tasks.

smaller compared to the FFT, which may be due to fewer trainable parameters and also illustrates the efficiency of the LFT. To gain a deeper understanding of the efficiency of LFT in comparison to FFT, we visually analyze the similarity between fine-tuned eigenvectors and pre-trained eigenvectors in Figure 4(c) and Figure 4(d). The results indicate that FFT updates exhibit a higher degree of randomness, whereas LFT updates align closely with the pre-training direction. Typically, pre-trained models offer a strong starting point for downstream tasks, and LFT's alignment with this pre-existing knowledge enables it to leverage learned information effectively. Consequently, this alignment eases the optimization process, elucidating why LFT demands less computational effort while still achieving effective fine-tuning of large models.

Empirical Study on Factors Influencing Lightweight Fine-tuning. To gain insights into the factors influencing the performance of PLMs. We conduct two empirical observations using BERT, ChildTuning, MPOP, and MaskedBERT. First, Table 4 shows that LFT on the proper component can even outperform FFT results (*e.g.,*, using FFN1 for SST-2 and attention for MRPC in MPOP). Second, in Table 5 the results on MRPC and SST-2 show that the best score typically appears after some FFT steps at the beginning of the fine-tuning process (*i.e.,* 40% FFT steps ratio). This illustrates the importance of the need to calibrate the start time of LFT.

Hyper-parameter Analysis. We conducted a sensitivity analysis of our approach with respect to two hyperparameters: the SRS threshold (δ) and the learning rate. Regarding the SRS threshold, we observed in Figure 5(a) that the optimal threshold for all downstream tasks consistently falls within the narrow optimal range (*i.e.,* from 0.45 to 0.6). To further investigate the impact of δ, we explored a range of options (0.4, 0.45, 0.5, 0.55, 0.6) and analyzed the results, as shown in Figure 5(b). The stability observed across most tasks demonstrates the robustness of our method to different threshold values. In terms of the learning rate, we performed experiments using different learning rates and evaluated the results on the SST-2, MRPC, and RTE tasks. The performance of our approach, as seen in Table 6, exhibited stable results with negligible variations in relation to the learning rate.

Models	Model Components FFN1	FFN2	Att	SST-2	MRPC
FFT	□	□	□	93.4	90.7
ChildTuning	✓	□	□	93.9	90.2
	□	✓	□	**94.2**	89.4
	□	□	✓	93.4	89.8
	✓	✓	✓	93.4	**90.3**
MPOP	✓	□	□	**94.5**	90.6
	□	✓	□	93.9	90.1
	□	□	✓	94.3	**90.9**
	✓	✓	✓	93.5	89.9
MaskBERT	✓	□	□	92.2	84.8
	□	✓	□	92.4	84.1
	□	□	✓	**92.5**	**85.3**
	✓	✓	✓	92.4	84.6

Table 4: Performance variations among different model components. We denote "✓" as the component using LFT strategy (*i.e.,* ChildTuning, MPOP and MaskBERT) while "□" uses FFT.

Models	FFT steps ratio	SST-2	MRPC
FFT	100%	93.4	90.7
ChildTuning	0%	93.4	90.3
	40%	**93.5**	**91.8**
	80%	92.9	91.3
MPOP	0%	93.5	89.9
	40%	**93.7**	**91.8**
	80%	93.3	91.4
MaskBERT	0%	92.4	84.6
	40%	**92.7**	**86.3**
	80%	92.5	86.0

Table 5: Performance variations with respect to optimization state. We denote the "FFT steps ratio" as the ratio of FFT steps at the beginning (in percent).

Learning Rate	5e-6	1e-5	3e-5	5e-5	1e-4
SST-2 (Acc.)	91.1	91.6	91.6	90.6	91.5
MRPC (F1)	88.3	87.6	87.4	87.8	86.3
RTE (Acc.)	57.4	57.4	59.9	59.6	60.7

Table 6: Comparison of different learning rates on $\text{BERT}_{\text{base}}$ (in percent).

5. Conclusion

This work proposes a method to choose the appropriate time to start lightweight fine-tuning for PLM components during full-parameter fine-tuning. To achieve this, we propose a metric, SRS, based on stable rank to assess the generalizability of the PLM and start lightweight fine-tuning when SRS suggests it yields better generalization behavior. Specifically, we propose a dynamic calibration (*i.e.,* CaliSr-D) for the overall model as well as a structure-aware calibration (*i.e.,* CaliSr-S) for individual components. Additionally, we provide insights into the low computational cost of lightweight fine-tuning by demonstrating that it primarily adjusts the distribution of eigenvalues rather than their magnitude. Finally, our approach is demonstrated to be effective through extensive experiments. We hope the CaliSr approach might benefit the understanding of the inner mechanism of lightweight fine-tuning.

6. Ethics Statement

Lightweight fine-tuning simplifies training for downstream tasks, but it may lead to the usage of biased datasets, particularly those with undesirable social consequences. These issues warrant more investigation and attention when expanding on this work and fine-tuning downstream tasks.

7. Acknowledgments

This work was partially supported by National Natural Science Foundation of China under Grant No. 62222215, 62206299 and 62376275. Xin Zhao is the corresponding author.

8. Bibliographical 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, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 7319–7328. Association for Computational Linguistics.

Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. 2017. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6240–6249.

Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. 2022. Revisiting parameter-efficient tuning: Are we really there yet? *CoRR*, abs/2202.07962.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained BERT networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Fahim Dalvi, Hassan Sajjad, Nadir Durrani, and Yonatan Belinkov. 2020. Analyzing redundancy in pretrained transformer models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4908–4926. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*.

Ze-Feng Gao, Song Cheng, Rong-Qiang He, ZY Xie, Hui-Hai Zhao, Zhong-Yi Lu, and Tao Xiang. 2020. Compressing deep neural networks by matrix product operators. *Physical Review Research*, 2(2):023300.

Ze-Feng Gao, Peiyu Liu, Wayne Xin Zhao, Zhong-Yi Lu, and Ji-Rong Wen. 2022. Parameter-efficient mixture-of-experts architecture for pre-trained language models. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3263–3273, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Ze-Feng Gao, Kun Zhou, Peiyu Liu, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Small pre-trained language models can be fine-tuned as large models via over-parameterization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3819–3834, Toronto, Canada. Association for Computational Linguistics.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4884–4896. Association for Computational Linguistics.

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

ER Henry and J Hofrichter. 1992. [8] singular value decomposition: Application to analysis of experimental data. *Methods in enzymology*, 210:129–192.

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 *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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

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 *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Ashish Khetan and Zohar S. Karnin. 2020. schubert: Optimizing elements of BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2807–2818. Association for Computational Linguistics.

Mikhail Khodak, Neil Tenenholtz, Lester Mackey, and Nicolo Fusi. 2021. Initialization and regularization of factorized neural layers. *arXiv preprint arXiv:2105.01029*.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. 2018. Measuring the intrinsic dimension of objective landscapes. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Peiyu Liu, Ze-Feng Gao, Yushuo Chen, Xin Zhao, and Ji-Rong Wen. 2023a. Enhancing scalability of pre-trained language models via efficient parameter sharing. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13771–13785, Singapore. Association for Computational Linguistics.

Peiyu Liu, Ze-Feng Gao, Wayne Xin Zhao, Zhi-Yuan Xie, Zhong-Yi Lu, and Ji-Rong Wen. 2021a. Enabling lightweight fine-tuning for pre-trained language model compression based on matrix product operators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5388–5398. Association for Computational Linguistics.

Peiyu Liu, Zikang Liu, Ze-Feng Gao, Dawei Gao, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. 2023b. Do emergent abilities exist in quantized large language models: An empirical study. *arXiv preprint arXiv:2307.08072*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabsa. 2021. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv preprint arXiv:2110.07577*.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. 2018. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry P. Vetrov. 2015. Tensorizing neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 442–450.

Ivan V Oseledets. 2011. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317.

Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur. 2018. Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Interspeech*, pages 3743–3747.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 506–516.

Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. 2013. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6655–6659. IEEE.

Amartya Sanyal, Philip H. S. Torr, and Puneet K. Dokania. 2020. Stable rank normalization for improved generalization in neural networks and gans. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Xingwei Sun, Ze-Feng Gao, Zhong-Yi Lu, Junfeng Li, and Yonghong Yan. 2020. A model compression method with matrix product operators for speech enhancement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2837–2847.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Benyou Wang, Yuxin Ren, Lifeng Shang, Xin Jiang, and Qun Liu. 2021. Exploring extreme parameter compression for pre-trained language models. In *International Conference on Learning Representations*.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. *CoRR*, abs/2204.00408.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9514–9528. Association for Computational Linguistics.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *CoRR*, abs/2106.10199.

Zhengyan Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2021. Moefication: Conditional computation of transformer models for efficient inference. *arXiv preprint arXiv:2110.01786*.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2226–2241. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. 2021. Taming sparsely activated transformer with stochastic experts. *CoRR*, abs/2110.04260.

9. Language Resource References

Alex Wang and Amanpreet Singh and Julian Michael and Felix Hill and Omer Levy and Samuel R. Bowman. 2019. *GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding*. OpenReview.net.