HiFi: High-Information Attention Heads Hold for Parameter-Efficient Model Adaptation

Anchun Gui and Han Xiao[*]
Department of Artificial Intelligence
School of Informatics, Xiamen University
anchungui@stu.xmu.edu.cn, bookman@xmu.edu.cn

Abstract

To fully leverage the advantages of large-scale pre-trained language models (PLMs) on downstream tasks, it has become a ubiquitous adaptation paradigm to fine-tune the entire parameters of PLMs. However, this paradigm poses issues of inefficient updating and resource over-consuming for fine-tuning in data-scarce and resource-limited scenarios, because of the large scale of parameters in PLMs. To alleviate these concerns, in this paper, we propose a parameter-efficient fine-tuning method HiFi, that is, only the highly informative and strongly correlated attention heads for the specific task are fine-tuned. To search for those significant attention heads, we develop a novel framework to analyze the effectiveness of heads. Specifically, we first model the relationship between heads into a graph from two perspectives of information richness and correlation, and then apply PageRank algorithm to determine the relative importance of each head. Extensive experiments on the GLUE benchmark demonstrate the effectiveness of our method, and show that HiFi obtains state-of-the-art performance over the prior baselines.

1 Introduction

Recently large-scale pre-trained language models (PLMs) have triggered a technological revolution in natural language processing (NLP), as the satisfactory performance could be achieved by fully fine-tuning parameters of PLMs (Devlin et al., 2019; Brown et al., 2020; Wei et al., 2021). In data-scarce and resource-limited scenarios, however, this methodology poses several concerns. On one hand, full fine-tuning leads to inefficient updating and catastrophic forgetting issues when the training set is insufficient (Houlsby et al., 2019; Wang et al., 2021); on the other hand, this approach has to duplicate a modified copy of full parameters per task, which presents the challenge of resource over-consuming for the limited storage infrastructure.

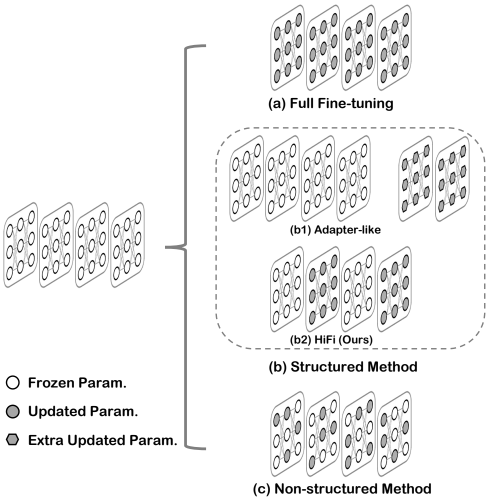

Figure 1: Comparison of diverse fine-tuning paradigms. For an example of a set of head weights, full fine-tuning updates all weights, whereas the non-structured methods randomly fine-tune a subset of parameters. For the structured methods, Adapter-like models update the extra weights, while our proposed HiFi selects several vital heads for fine-tuning.

Parameter-efficient fine-tuning (PEFT), as an alternative paradigm, has attracted more attention recently (He et al., 2022; Ding et al., 2022). Compared to full fine-tuning, PEFT only fine-tunes the minority of the original parameters (or extra introduced parameters) instead of the entire parameters of PLMs, as shown in Fig. 1. Although current PEFTs effectively decrease the proportion of trainable parameters, these methods also lead to varying aspects of concerns. For instance, Adapter (Houlsby et al., 2019) not only breaks the model structure by introducing additional parameters but also causes inference delays (Hu et al., 2022). We compare the representative models in Tab. 1.

Motivated by these issues, in this paper, we propose HiFi, a novel PEFT method by fine-tuning the relatively significant heads in multi-head attention

[*] Corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 8521–8537
July 9-14, 2023 ©2023 Association for Computational Linguistics

(MHA) modules, where we assume that our PLMs are based on Transformer (Vaswani et al., 2017) architecture and MHA is chosen since it plays a crucial role in Transformer according to recent studies (Voita et al., 2019; Baan et al., 2019; Michel et al., 2019). There are several intractable challenges to implementing our proposed method.

How to measure the individual importance of a head? The core question of HiFi is to select relative importance heads in MHA. Toward this end, we first analyze the importance of a single head. Specifically, we decompose the output of each head by singular value decomposition (SVD) to obtain a sequence of singular values in descending order, where if the cumulation of top-t terms account for a percentage threshold (e.g., 90%), the index t is treated as a measurement of information richness of the head.

How to measure the relative importance between heads? Given that the collaboration among multiple heads achieves success (Kovaleva et al., 2019; Clark et al., 2019), the head-to-head correlation also shall be considered. Therefore, we calculate the covariance matrix between the outputs of heads to measure the correlation across heads. To further work out those highly informative and strongly correlated heads, we model the relationship between heads into a graph based on their information richness and correlation, and then derive the relative importance of each head by PageRank algorithm (Page et al., 1999). We illustrate our method overview in Fig. 2.

To verify the effectiveness of our proposed method, we conduct extensive experiments on the GLUE benchmark (Wang et al., 2018), in both full-shot[1] and few-shot settings. The results show that our model HiFi gains state-of-the-art performance against strong baselines. For example, full fine-tuning achieves the average score of 82.0% in the full-shot setting, while HiFi obtains superior performance (82.3%).

To summarize, our contributions are as follows:

- We develop a novel framework for analyzing the relative importance of weights, and empirically demonstrate its robustness under diverse experimental settings.

- Based on this framework, we propose a simple yet effective PEFT method, HiFi. Our method fulfills the performance requirement without introducing additional concerns compared to the previous baselines, as shown in Tab. 1.

Method	Extra Param.	Corrupt Struc.	Infer. Delay	Store Cons.
Full-FT	✗	✗	✗	✓
Adapter	✓	✓	✓	✗
LoRA	✓	✓	✗	✗
Prompt-Tuning	✓	✗	✓	✗
Diff-Pruning	✓	✗	✗	✓
Child-Tuning	✗	✗	✗	✓
HiFi (Ours)	✗	✗	✗	✗

Table 1: Compared with related methods, our proposed HiFi does not raise additional concerns. “Extra Param.”: introduces new trainable parameters apart from the original parameters. “Corrupt Struc.”: corrupts the original structure of the model. “Infer. Delay”: causes inference delay. “Store Cons.”: saves all parameters per task or the updated parameters are not convenient to store[2].

- Our proposed HiFi outperforms the current strong counterparts on the GLUE benchmark, in both full-shot and few-shot settings. We also verify the effectiveness of our methodology by abundant analytical experiments.

2 Related Work

The existent research on PEFT can be generally divided into two folds: structured and non-structured methods. For the structured methods, the updated parameters are modularized (i.e., the parameters from the particular weight blocks are tuned), while the location of those updated parameters in the non-structured methods is irregular, as shown in Fig. 1.

Structured Methods. There are two types of updatable blocks. (i) Extra introduced blocks. For example, Adapter (Houlsby et al., 2019) inserts compact modules into PLMs, and LoRA (Hu et al., 2022) introduces two low-rank MLPs into the query and key weights in MHA. Similar models include Compacter/Compacter++ (Mahabadi et al., 2021), AdapterDrop (Rücklé et al., 2021), AdapterBias (Fu et al., 2022), and MAM (He et al., 2022). In addition, Prompt-Tuning (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2022) is also a popular research direction via prepending a sequence of continuous learnable vectors to the input. Generally, we refer these models to Adapter-like methods in Fig. 1, because they introduce additional learnable

[1] It refers to vanilla fine-tuning setting, see Sec. 4.1.

[2] For the non-structured methods (e.g., Diff-Pruning (Guo et al., 2021), Child-Tuning (Xu et al., 2021)), since the position of updated parameters is unordered as shown in Fig. 1(c), we cannot directly save them in a weight matrix.

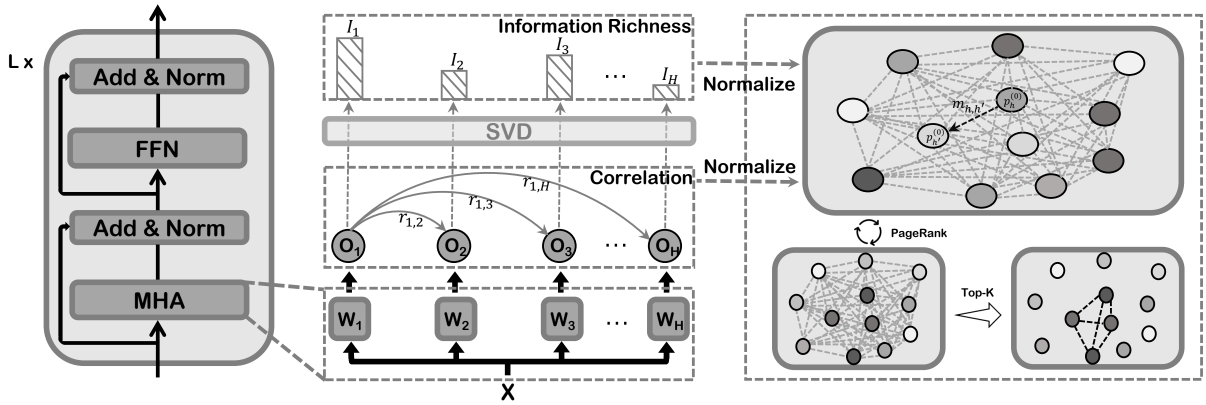

Figure 2: An overview of our method. For each layer, we first calculate the information richness of a single head and the correlation between heads, then construct a graph by normalizing our proposed metrics. For a specific downstream task, we search for the relative significant heads for fine-tuning using PageRank algorithm. The darker ball on the right figure indicates more important head.

parameters apart from the original model parameters. (ii) Internal original blocks. BitFit (Ben-Zaken et al., 2022), for instance, fine-tunes the all bias terms of PLMs on downstream tasks. The distinctions between our proposed HiFi and BitFit are as follows: 1) we tune the attention heads rather than bias, given that the heads act a significant role in Transformer (Voita et al., 2019; Baan et al., 2019); 2) our selected heads are task-specific, while BitFit does not consider the information of downstream tasks.

Non-structured Methods. The core question in this direction is that: how to identify a sparse subnetwork based on the importance of each parameter in PLMs? Child-Tuning (Xu et al., 2021) calculates the Fisher Information Matrix (FIM) of parameters to determine a "child" network in the original network. Diff-Pruning (Guo et al., 2021) learns a perturbation variable per parameter by L_0 norm constraint so that the updatable parameters are selected automatically. Similar work includes Ansell et al. (2022).

3 Methodology

3.1 Notations

Supposing the PLM consists of L encoder layers, and MHA has H attention heads and the corresponding weights[3] are $W_h^Q, W_h^K, W_h^V \in \mathbb{R}^{D \times D'}, h \in \{1, 2, \cdots, H\}$, where D refers to the hidden representation dimensions and $D' = \frac{D}{H}$. Let the weight set of the h-th head be $W_h = \{W_h^Q, W_h^K, W_h^V\}$. For a sample x from the data distribution $\mathcal{D}$, $O_h(x) \in \mathbb{R}^{S \times D'}$ represents the output of x through the h-th head, where S indicates the sequence length of x.

Besides, in our work, we notate that $g(\cdot)$ measures the individual importance of a head and $r(\cdot, \cdot)$ indicates the correlation between two heads. The design of metrics should satisfy the following principles: task-relevant and robustness, because we expect $g(\cdot), r(\cdot, \cdot)$ to capture the intrinsic properties of heads on a range of downstream tasks.

3.2 Information Richness

Inspired by leveraging the feature map to measure the importance of the corresponding convolution kernel in computer vision (Lin et al., 2020), we here treat O_h equivalently as the "feature map" in our scenario. Specifically, rather than focusing solely on W_h, we analyze its output O_h to reflect the importance of W_h and give the following definition:

$$g(W_h) = \mathbb{E}_{x \sim \mathcal{D}} \left[g(O_h(x)) \right] \tag{1}$$

By calculating the expectation of O_h with respect to x, we can measure the importance of the h-th head based on a particular task. The more critical the head, intuitively, the richer the corresponding output should be. To this end, we characterize this property from the perspective of singular value decomposition (SVD). Specifically, we perform the SVD on the output, i.e., $O_h(x) = U \Sigma V^\top = U \operatorname{diag}(\sigma_1, \sigma_2, \cdots, \sigma_T) V^\top$, where $T = \min\{S, D'\}, \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_T$.

Based on the properties of SVD, we are aware that if the sequence of singular values $\{\sigma_t\}$ decays

[3] We ignore the bias terms for simplicity.

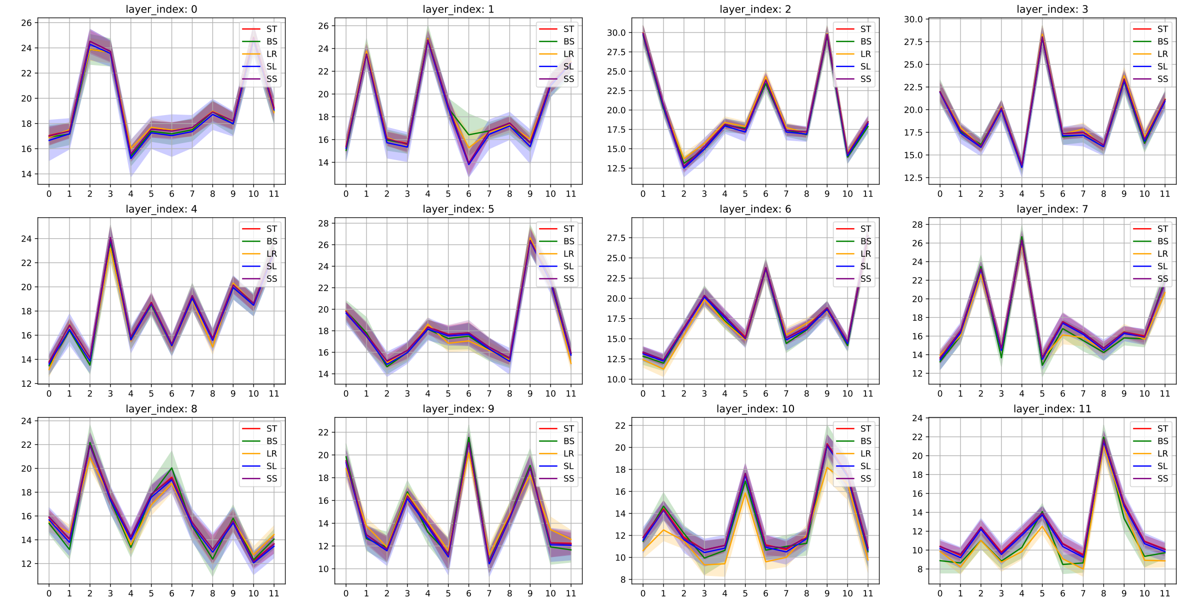

Figure 3: The robustness of information richness (I_h). Compared to the standard setting of **ST**, **BS** reduces the batch size, **LR** increases the learning rate, **SL** reduces the sequence length, and **SS** increases the sample size. In each subfigure, x-axis and y-axis represent the index of heads and corresponding I_h. The solid line and shading area refer to the mean and standard deviation, respectively. See Sec. 5.1 for detailed experimental settings.

slower, it means that O_h is informative and contains more meaningful principal components. Therefore, given the specific task information ($x \sim \mathcal{D}$), we define the information richness of an attention head as $I_h(W_h|x)$:

$$I_h(W_h|x) = \underset{t}{argmin} \left\{ \frac{\sum_{i=1}^{t} \sigma_i}{\sum_{j=1}^{T} \sigma_j} \geq \xi \right\} \tag{2}$$

where ξ is a hyperparameter threshold, we set $\xi = 0.9$ in our experiments. Note that Eq. (1) requires solving the expectation on data x. In practice, we approximate the solution using the Monte-Carlo algorithm:

$$\begin{aligned} \mathbb{E}_{x\sim\mathcal{D}}\left[g(O_h(x))\right] &= \mathbb{E}_{x\sim\mathcal{D}}\left[I_h(W_h|x)\right] \\ &\approx \frac{1}{n}\sum_{i=1}^{n} I_h(W_h|x_i) \end{aligned} \tag{3}$$

Although this operation is expensive when the amount of training data is large, we fortunately find that the stable results can be obtained using a small n (e.g., 300) in actual experiments. Besides, this metric also remains robust under diverse settings as shown in Fig. 3.

3.3 Correlation

Applying a similar idea from the solution of I_h, the correlation between weights is transformed into the corresponding outputs. We define the correlation $r(W_h, W_{h'})$ between two heads (h, h') as:

$$r(W_h, W_{h'}) = \mathbb{E}_{x\sim\mathcal{D}}\left[r(O_h, O_{h'}|x)\right] \tag{4}$$

where the outputs $O_h, O_{h'} \in \mathbb{R}^{S\times D'}$. To calculate $r(O_h, O_{h'}|x)$, we first average O_h over the sequence axis, i.e., $O'_h = \frac{1}{S}\sum_{s=1}^{S} O_h(s,:)$, where $O_h(s,:) \in \mathbb{R}^{D'}$ refers to the hidden representation of the s-th token in the sequence.

Next, the correlation between two heads (h, h') is computed by the covariance:

$$r(O'_h, O'_{h'}|x) = \left|\mathrm{cov}(O'_h(x), O'_{h'}(x))\right| \tag{5}$$

Here, we are focusing on the degree of correlation, where the correlation for strong positive and negative should be considered equally. Hence, we put the absolute operation on Eq. (5). We apply the unbiased estimation of covariance:

$$\mathrm{cov}(O'_h, O'_{h'}) = \frac{\sum_{d=1}^{D'}(o_{h,d} - \bar{o}_h)(o_{h',d} - \bar{o}_{h'})}{D' - 1} \tag{6}$$

where $o_{h,d}$ and $o_{h',d}$ represent the d-th element in O'_h and $O'_{h'}$, while $\bar{o}_h$ and $\bar{o}_{h'}$ indicate the average of O'_h and $O'_{h'}$, respectively. Thus, the correlation matrix of heads is defined as $R = [r_{h,h'}]_{H\times H}$, the

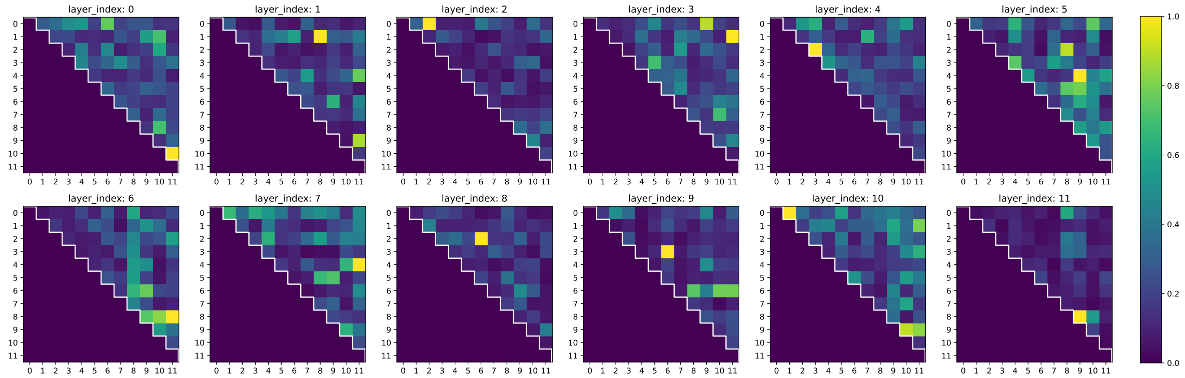

Figure 4: The head-to-head correlation ($r_{h,h'}$). In each subfigure, both x-axis and y-axis represent the index of heads, and the value of correlation is normalized. See Sec. 5.1 for detailed experimental settings.

entry of which is $r_{h,h'} \doteq \mathbb{E}_{x\sim\mathcal{D}}\left[r(O_h, O_{h'}|x)\right]$ and $r_{h,h} = 0$. For the calculation of Eq. (4), we also adopt the Monte-Carlo algorithm to approximate the solution:

$$\mathbb{E}_{x\sim\mathcal{D}}\left[r(O_h, O_{h'}|x)\right] \approx \frac{1}{n}\sum_{i=1}^{n} r(O_h, O_{h'}|x_i) \tag{7}$$

We show the head-to-head correlation heatmap in Fig. 4. In addition, we report more comparisons in appendix A.4 to illustrate its robustness under different experimental settings.

3.4 Joint Optimization

To summarize, we can obtain I_h, which measures the information richness of a head, and R, which indicates the correlation matrix between heads. To determine the relative importance of each head, we first model the relationship between heads as a directed fully-connected graph, as shown in Fig. 2. In this graph, the initial probability $p_h^{(0)}$ per node (i.e., head) is defined as:

$$p_h^{(0)} = \frac{I_h}{\sum_{h'=1}^{H} I_{h'}} \tag{8}$$

Then, we define $m_{h,h'}$, the probability of moving from node h to another node h', as:

$$m_{h,h'} = \frac{r_{h,h'}}{\sum_{h''\neq h}^{H} r_{h,h''}} \tag{9}$$

Hence, we can obtain the initial probability vector $P^{(0)} = [p_1^{(0)}, p_2^{(0)}, \cdots, p_H^{(0)}]^\top$ and the state transition probability matrix $M = [m_{h,h'}]_{H\times H}$. Given that H is generally small in practice (e.g., $H = 16$ in BERT$_{\text{LARGE}}$), we employ the iterative method of PageRank (Page et al., 1999) to search for the optimal solution:

$$P^{(t+1)} = dMP^{(t)} + \frac{1-d}{H}\mathbb{I} \tag{10}$$

where d refers to the damping factor, and $\mathbb{I}$ is the H-dimensional vector with all elements of 1. From the perspective of PageRank (Page et al., 1999), when the Markov chain reaches stationary distribution, the PageRank value p_h^* per node in the graph can be obtained: $P^* = \lim_{t\to\infty} P^{(t+1)} = [p_1^*, p_2^*, \cdots, p_H^*]^\top$. Finally, we utilize p_h^* to evaluate the relative importance of the h-th head, and then take the top-k heads for fine-tuning. The detailed algorithm procedures are summarized in appendix A.5.

4 Experiments

4.1 Setup

Datasets & Evaluation Protocol. Following the previous setting (Houlsby et al., 2019), we evaluate our method on the GLUE benchmark (Wang et al., 2018), which consists of eight datasets (i.e., CoLA, SST-2, MPRC, QQP, STS-B, MNLI, QNLI, and RTE). See A.1 in appendix for detailed description per dataset. In addition, Matthew's (Matt.) and Spearman's (Spea.) correlation coefficient are used to test CoLA and STS-B, respectively. MRPC and QQP are measured by F1 score. As for other datasets, the accuracy (Acc.) metric is applied.

Full-shot Learning. According to the vanilla evaluation procedure, we utilize the datasets library[4] to load the complete dataset for training, and then save the best checkpoint based on the performance on validation set, and finally report the

[4] https://github.com/huggingface/datasets

Model	QNLI (Acc.)	SST-2 (Acc.)	MNLI$_m$ (Acc.)	MNLI$_{mm}$ (Acc.)	CoLA (Matt.)	MRPC (F1)	STS-B (Spea.)	RTE (Acc.)	QQP (F1)	Avg.
Full-shot Learning										
Full-FT†	93.4	94.1	**86.7**	**86.0**	59.6	88.9	86.6	71.2	71.7	82.0
Diff-Pruning†	93.3	94.1	86.4	**86.0**	**61.1**	**89.7**	86.0	70.6	71.1	82.0
Child-Tuning	92.6$_{\pm 0.3}$	94.2$_{\pm 0.5}$	86.1$_{\pm 0.6}$	85.2$_{\pm 0.4}$	59.2$_{\pm 0.5}$	88.1$_{\pm 0.8}$	85.3$_{\pm 0.5}$	71.2$_{\pm 0.5}$	71.3$_{\pm 0.3}$	81.5
Adapter†	90.7	94.0	84.9	85.1	59.5	89.5	86.9	71.5	71.8	81.5
BitFit†	92.0	94.2	84.5	84.8	59.7	88.9	85.5	**72.0**	70.5	81.3
LoRA	91.2$_{\pm 0.5}$	93.2$_{\pm 0.3}$	84.2$_{\pm 0.7}$	84.1$_{\pm 0.5}$	60.2$_{\pm 0.9}$	88.8$_{\pm 0.7}$	85.9$_{\pm 0.2}$	70.3$_{\pm 0.3}$	71.0$_{\pm 0.5}$	81.0
Compacter	91.5$_{\pm 0.2}$	93.6$_{\pm 0.4}$	85.3$_{\pm 0.5}$	84.9$_{\pm 0.3}$	58.6$_{\pm 0.6}$	87.9$_{\pm 1.0}$	86.6$_{\pm 0.6}$	69.7$_{\pm 0.4}$	**71.8**$_{\pm 0.2}$	81.1
Prefix-Tuning	92.2$_{\pm 0.5}$	**94.3**$_{\pm 0.3}$	84.2$_{\pm 0.3}$	84.0$_{\pm 0.4}$	58.4$_{\pm 0.8}$	88.2$_{\pm 0.5}$	85.7$_{\pm 0.3}$	71.3$_{\pm 0.2}$	69.7$_{\pm 0.6}$	80.9
HiFi$_{\text{mid-top}}$	92.7$_{\pm 0.2}$	93.9$_{\pm 0.5}$	85.8$_{\pm 0.3}$	85.5$_{\pm 0.6}$	59.1$_{\pm 0.9}$	88.6$_{\pm 0.6}$	86.8$_{\pm 0.2}$	70.8$_{\pm 0.5}$	71.5$_{\pm 0.8}$	81.6
HiFi$_{\text{layer-wise}}$	**93.5**$_{\pm 0.6}$	**94.3**$_{\pm 0.3}$	85.9$_{\pm 0.7}$	85.8$_{\pm 0.4}$	60.4$_{\pm 0.7}$	**89.7**$_{\pm 0.4}$	**87.2**$_{\pm 0.3}$	71.5$_{\pm 0.2}$	**72.0**$_{\pm 0.4}$	**82.3**
Few-shot Learning										
Full-FT	67.9$_{\pm 3.8}$	72.4$_{\pm 6.7}$	44.1$_{\pm 5.4}$	45.1$_{\pm 5.6}$	**28.5**$_{\pm 5.5}$	73.3$_{\pm 6.3}$	-	54.5$_{\pm 4.5}$	59.5$_{\pm 4.8}$	55.7
Diff-Pruning	63.1$_{\pm 1.3}$	74.0$_{\pm 5.4}$	42.5$_{\pm 6.1}$	40.6$_{\pm 5.2}$	21.1$_{\pm 8.9}$	72.7$_{\pm 2.8}$	-	53.5$_{\pm 3.1}$	57.8$_{\pm 4.5}$	53.2
Child-Tuning	65.8$_{\pm 2.2}$	76.1$_{\pm 4.6}$	40.7$_{\pm 4.3}$	41.4$_{\pm 3.7}$	24.7$_{\pm 7.5}$	73.1$_{\pm 3.9}$	-	52.5$_{\pm 4.2}$	58.3$_{\pm 3.8}$	54.1
Adapter	67.2$_{\pm 3.3}$	78.3$_{\pm 4.6}$	42.2$_{\pm 5.0}$	44.5$_{\pm 5.7}$	26.6$_{\pm 4.2}$	73.0$_{\pm 7.6}$	-	55.0$_{\pm 2.2}$	59.2$_{\pm 2.9}$	55.8
BitFit	**70.3**$_{\pm 2.1}$	78.9$_{\pm 7.4}$	43.5$_{\pm 3.4}$	42.9$_{\pm 4.4}$	23.8$_{\pm 9.2}$	72.4$_{\pm 4.8}$	-	54.7$_{\pm 1.1}$	61.3$_{\pm 4.3}$	56.0
LoRA	68.8$_{\pm 1.9}$	**80.3**$_{\pm 5.2}$	41.3$_{\pm 2.7}$	42.9$_{\pm 3.0}$	27.1$_{\pm 7.2}$	75.8$_{\pm 5.9}$	-	55.2$_{\pm 1.9}$	60.5$_{\pm 6.1}$	56.5
Compacter	69.6$_{\pm 1.8}$	76.6$_{\pm 9.5}$	43.4$_{\pm 6.0}$	45.4$_{\pm 6.9}$	25.9$_{\pm 8.5}$	73.5$_{\pm 7.2}$	-	52.4$_{\pm 3.4}$	60.8$_{\pm 4.6}$	56.0
Prefix-Tuning	68.3$_{\pm 3.6}$	79.2$_{\pm 1.9}$	43.3$_{\pm 4.5}$	45.7$_{\pm 4.8}$	24.8$_{\pm 3.3}$	72.4$_{\pm 9.3}$	-	54.4$_{\pm 2.5}$	60.4$_{\pm 2.3}$	56.1
HiFi$_{\text{mid-top}}$	67.7$_{\pm 1.6}$	76.2$_{\pm 2.4}$	44.8$_{\pm 4.4}$	44.7$_{\pm 5.3}$	26.8$_{\pm 6.5}$	75.2$_{\pm 5.2}$	-	55.4$_{\pm 3.8}$	59.7$_{\pm 3.9}$	56.3
HiFi$_{\text{layer-wise}}$	68.5$_{\pm 2.7}$	76.6$_{\pm 3.5}$	**45.9**$_{\pm 5.7}$	**45.8**$_{\pm 6.3}$	**28.5**$_{\pm 5.2}$	**76.3**$_{\pm 3.5}$	-	**55.5**$_{\pm 3.5}$	**61.8**$_{\pm 4.1}$	**57.4**

Table 2: The performance on the GLUE benchmark. The results are averaged from three seeds in the full-shot learning[6], while five seeds are used in the few-shot learning to produce solid results. The subscript is the standard deviation. **Bold** and underline indicate the first and second best results in the corresponding regime. † refers to the results directly from their original paper, in which Full-FT is derived from Guo et al. (2021).

results on test set by submitting our predictions to the online evaluator[5].

Few-shot Learning. Following the setting of Sun et al. (2022), we randomly select 16 samples per class to construct 16-shot training set $\mathcal{D}_{\text{train}}$ and validation set $\mathcal{D}_{\text{val}}$ from the original training set, respectively. In addition, the original validation set is regarded as the test set $\mathcal{D}_{\text{test}}$, where $|\mathcal{D}_{\text{train}}| = |\mathcal{D}_{\text{val}}| \ll |\mathcal{D}_{\text{test}}|$. The STS-B dataset is excluded since it is a regressive task.

Baselines. To make a comprehensive comparison, we first select full fine-tuning (Full-FT) as a strong baseline, then choose five structured methods (i.e., Adapter (Houlsby et al., 2019), Compactor (Mahabadi et al., 2021), Prefix-Tuning (Li and Liang, 2021), LoRA (Hu et al., 2022) and BitFit (Ben-Zaken et al., 2022)), and two non-structured methods (i.e., Diff-Pruning (Guo et al., 2021), and Child-Tuning (Xu et al., 2021)) as the other baselines. See A.2 in appendix for more description of each baseline.

Models & Implementation. Given that, in pre-training, the lower layers of PLMs learn general semantic features, which might be universal across tasks (Houlsby et al., 2019; Rücklé et al., 2021). Therefore, two models are proposed here: **HiFi$_{\text{layer-wise}}$** fine-tunes the selected top-k heads at each layer; **HiFi$_{\text{mid-top}}$** only updates the layers from middle to top, while keeping the layers from bottom to middle frozen. k is set to 3 by default and its implications are further explored in Sec. 5.3. We leverage BERT$_{\text{LARGE}}$ as backbone and implement our models by Huggingface's Transformers library (Wolf et al., 2020). The off-the-shelf Adapter-Transformers library[7] (Pfeiffer et al., 2020) is utilized to perform the baselines. See A.3 in appendix for more detailed experimental configurations.

[5] `https://gluebenchmark.com/`

[6] The average score of † differs slightly from the original paper due to the different averaging method, e.g., BitFit first averages MNLI$_m$ and MNLI$_{mm}$, and then calculates the overall average score. Here, we directly calculate the average score of all datasets on GLUE, in a unified manner.

[7] `https://github.com/Adapter-Hub/adapter-transformers`

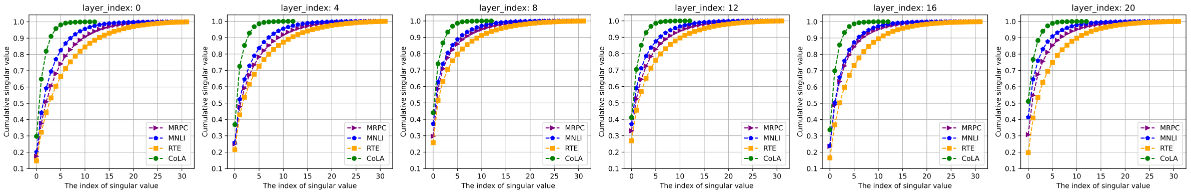

Figure 5: The effect of cumulative singular value on various datasets. We randomly select the output of a head in different layers for illustration.

Model	QNLI	SST-2	MNLI$_m$	MNLI$_{mm}$	CoLA	MRPC	STS-B	RTE	QQP	Avg.
HiFi	93.2	94.5	85.9	85.8	61.5	92.6	88.1	73.3	87.8	84.7
HiFi (w/o corr)	92.2	93.2	85.4	85.1	61.1	92.0	88.0	72.8	87.2	84.1
HiFi (w/o corr + inv)	91.8	93.5	84.8	84.6	60.8	89.1	87.1	72.0	86.9	83.4
HiFi (w/o info)	92.6	93.4	84.4	84.2	61.1	91.4	87.8	72.2	87.5	83.8
HiFi (page inv)	92.4	93.6	84.3	84.5	60.4	91.2	87.6	72.8	86.8	83.7
HiFi (random)	90.2	92.4	83.5	83.1	59.3	88.5	86.5	70.6	86.0	82.4

Table 3: Ablation experiments.

4.2 Results

In Tab. 2, we show the results of our models and baselines on the GLUE benchmark. From a wide range of comparisons, we can obtain that: (i) Under the full-shot setting, our proposed **HiFi$_{\text{layer-wise}}$** obtains the average of 82.3, which outperforms Full-FT (82.0). Meanwhile, **HiFi$_{\text{mid-top}}$** also achieves satisfactory performance (81.6), in the case of tuning only half of the trainable parameters compared to **HiFi$_{\text{layer-wise}}$**. Besides, the non-structured methods achieve better performance than the prior structured methods and are closer to the results of Full-FT, but none of them exceed Full-FT on average. (ii) In the few-shot setting, the structured methods have significant advantages over Full-FT and non-structured methods. Our models (including the previous structured methods) achieve higher average score than Full-FT, while the non-structured methods are substantially lower than Full-FT[8].

In short, our proposed HiFi bridges the gap between structured and non-structured methods in the full-shot setting, while maintaining a significant lead in the few-shot setting. On one hand, as we have discussed before, those heads we selected are vital for downstream tasks and could greatly influence model decisions; on the other hand, it acts as a regularizer to fine-tune only a subset of the full parameters, which boosts the generalization capacity. We further verify this hypothesis from the perspective of loss landscape in Sec. 5.4.

5 Analysis[9]

5.1 Robustness of Metrics

In this subsection, we study the following question: whether we designed metrics ($I_h, r_{h,h'}$) are robust enough under diverse settings? To this end, we take BERT$_{\text{BASE}}$ (12 layers) as an example to conduct experiments on MRPC[10], in terms of batch size, sequence length, sample size (i.e., n) and learning rate. Specifically, **ST** is a standard baseline, where the batch size is 32, the learning rate is $2e^{-5}$, the maximum sequence length is 128 and the sample size is 300. Compared to **ST**, **BS** reduces the batch size to 16, **LR** increases the learning rate to $5e^{-5}$, **SL** reduces the maximum sequence length to 64, and **SS** increases the number of samples to 1000.

The results are shown in Fig. 3, 4 (partially in appendix A.4), we can draw the following findings: (i) For the information richness (I_h), even if the experimental settings are diverse, I_h almost remains consistent (e.g., in $2, 4, 6$ layers). Note that I_h will be normalized by Eq. (8), the absolute ranking of I_h is inessential (e.g., in $10, 11$ layers, the curves are slightly moved up and down), as long as the relative ranking across heads remains stable. (ii) For the correlation ($r_{h,h'}$), the head-to-head correlation in the heatmap is varying across layers, which means that this metric has good distinguishability.

[8] For an example of Diff-Pruning, the reason is that it probably fails to optimize the perturbation variable for each original parameter with such small training samples.

[9] In this section, we conduct the experiments based on HiFi$_{\text{layer-wise}}$, unless otherwise specified.

[10] In fact, the consistent observation emerges on BERT$_{\text{LARGE}}$ and different datasets.

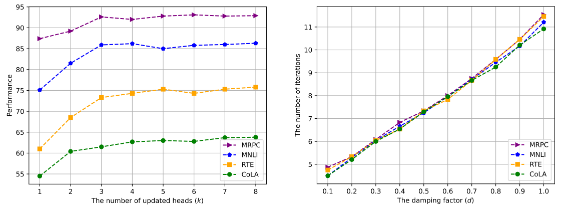

Figure 6: The effect of the number of selected heads k (Left) and the damping factor d (Right).

In addition, although the correlation region varies slightly in diverse settings, the strongly correlated heads are almost unchanged[11]. This demonstrates that I_h and $r_{h,h'}$ have good robustness and capture some essential characteristics of heads.

5.2 Effectiveness of Methods

Here, we focus on investigating the possible questions: Q1: Does the correlation ($r_{h,h'}$) between heads really matter? Q2: Are the higher information richness (I_h) of heads more important for the model? Q3: Is it enough to only take the correlation ($r_{h,h'}$) into consideration, while ignoring the information richness (I_h)? Q4: Does PageRank algorithm really work?

To answer the above questions, we design the following experiments for verification: (i) For Q1, we exclude the correlation ($r_{h,h'}$) compared to the baseline, and update k heads corresponding the top information richness. This experiment denotes **HiFi (w/o corr)**. (ii) For Q2, in contrast to Q1, we merely update k heads with the lowest information richness, without taking $r_{h,h'}$ into account. This experiment denotes **HiFi (w/o corr + inv)**. (iii) For Q3, the information richness is not included and only k heads with the strongest correlation are updated. This experiment denotes **HiFi (w/o info)**. (iv) For Q4, compared to the baseline, we inversely select k heads with the lowest PageRank value. This experiment denotes **HiFi (page inv)**. In addition, to establish the effectiveness of our selected heads, we also compare with a baseline that randomly tunes k heads in each layer. This experiment denotes **HiFi (random)**. Here, the experiments are conducted on the validation sets of GLUE, and the comparison results are shown in Tab. 3. We observe that the optimal performance is only obtained when the information richness (I_h) and correlation ($r_{h,h'}$) are jointly addressed by PageRank algorithm.

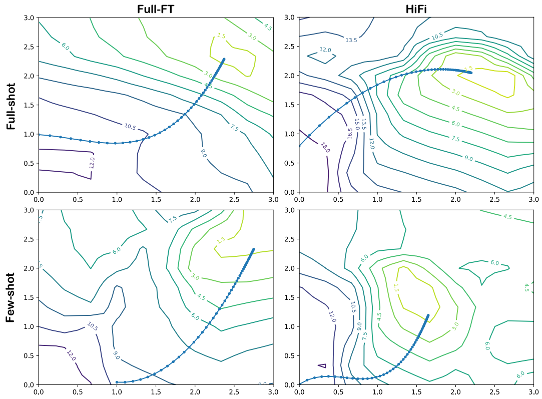

Figure 7: Visualization of loss contours and training trajectories of Full-FT (first column) and HiFi (second column), in both full-shot and few-shot scenarios.

5.3 Influence of Hyperparameters

In this subsection, we probe the following hyperparameters: the number of selected heads (k), the proportion of the top-t singular values (ξ) and the damping factor (d), based on MPRC, CoLA, RTE and MNLI, respectively. (i) k is a key factor to control the ratio of trainable parameters over the all parameters. In Fig. 6, the optimal value of k is around 3 and the corresponding ratio is 4.2%. If the mid-top strategy is adopted, the ratio can be further reduced to 2.1%[12]. (ii) The curves have greater curvature when the y-axis reaches around 0.9, and the growth subsequently becomes slow as shown in Fig. 5, which indicates that the top-t principal components are highly informative. Therefore, $\xi = 0.9$ is a good boundary. (iii) In Fig. 6, the number of iterations grows as d increases. Nevertheless, we find that the PageRank algorithm is fast (less than 0.4 second) in practical experiments. More analysis about efficiency in appendix A.6. Besides, the convergence consistency of PageRank (Page et al., 1999) also guarantees the stability of our results, once $P^{(0)}$ and M had been obtained.

5.4 Perspective of Loss Landscape

To further understand the effectiveness of HiFi compared to Full-FT, we visualize the training trajectories along with loss landscapes through tools (Li et al., 2018; Hao et al., 2019) on MRPC. As shown in Fig. 7, the loss contour of HiFi is smoother/flatter than Full-FT in both full-shot and few-shot settings. A smoother/flatter loss surface is often believed to

[11] Compare Fig. 4 with Fig. 8-11 in appendix, where each figure is produced in a setting.

[12] In fact, this ratio can be further decreased as the model scale increases because most parameters of Transformer block are concentrated in the feedforward (66.7%), while MHA is only 33.3%, in which we fine-tune several heads.

indicate enhanced standard and robust generalization (He et al., 2019; Wu et al., 2020), where an intuitive understanding is that if optimum basin is flat enough, it is possible to avoid the model jumping out the well-generalized region. Therefore, this provides a possible perspective to explain that HiFi has a more robust generalization capability than Full-FT on downstream tasks.

6 Conclusion

In this paper, we propose a novel PEFT method HiFi by modeling the relationship between attention heads as a graph and then using the PageRank to determine the relative significant heads for fine-tuning. HiFi obtains state-of-the-art performance over the prior baselines on GLUE benchmark, in both full-shot and few-shot scenarios.

Acknowledgment

This project is supported by the Young Scientists Fund of the National Natural Science Foundation of China (Grant No. 62006201).

Limitations

In this work, we design a parameter-efficient fine-tuning method and demonstrate its effectiveness through extensive experiments. However, there are some limitations: (i) It may be suboptimal to select the same number of heads (k) in each layer. In future work, we will continue to explore how to design better selection strategies (e.g., an intuitive idea is that k should be layer-specific). (ii) So far, the models and datasets we tested all belong to NLP. Due to time and resource limitations, we do not evaluate the related models and datasets (e.g., ViT, CIFAR) in CV.

Ethics Statement

Our proposed HiFi can effectively reduce the resource consumption of PLMs during the fine-tuning phase, which helps to decrease carbon emissions, thus making the large-scale models more environmentally friendly and sustainable. In addition, all the models and datasets used in our experiments are publicly available and have not been reported to carry social bias against any sensitive attributes, and the proposed approach would not explicitly introduce new negative societal impacts.

References

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.

Joris Baan, Maartje ter Hoeve, Marlies van der Wees, Anne Schuth, and Maarten de Rijke. 2019. Understanding Multi-Head Attention in Abstractive Summarization. *arXiv:1911.03898 [cs]*.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

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, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2022. Delta Tuning: A Comprehensive Study of Parameter Efficient Methods for Pre-trained Language Models. *arXiv:2203.06904 [cs]*.

Chin-Lun Fu, Zih-Ching Chen, Yun-Ru Lee, and Hung-yi Lee. 2022. AdapterBias: Parameter-efficient Token-dependent Representation Shift for Adapters in NLP Tasks. *arXiv:2205.00305 [cs]*.

Demi Guo, Alexander 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 (Volume 1: Long Papers)*, pages 4884–4896.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. Visualizing and Understanding the Effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4143–4152.

Haowei He, Gao Huang, and Yang Yuan. 2019. Asymmetric Valleys: Beyond Sharp and Flat Local Minima. In *Advances in Neural Information Processing Systems*, volume 32.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a Unified View of Parameter-Efficient Transfer Learning. In *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 *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799.

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*.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the Dark Secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374.

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.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. 2018. Visualizing the Loss Landscape of Neural Nets. In *Advances in Neural Information Processing Systems*, volume 31.

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.

Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. 2020. HRank: Filter Pruning Using High-Rank Feature Map. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1526–1535.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient Low-Rank Hypercomplex Adapter Layers. In *Advances in Neural Information Processing Systems*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are Sixteen Heads Really Better than One? In *Advances in Neural Information Processing Systems*, volume 32.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bringing Order to the Web. http://ilpubs.stanford.edu:8090/422/.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A Framework for Adapting Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54.

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.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the Efficiency of Adapters in Transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-Box Tuning for Language-Model-as-a-Service. In *Proceedings of the 39th International Conference on Machine Learning*, pages 20841–20855.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. In *Proceedings of the*

57th Annual Meeting of the Association for Computational Linguistics, pages 5797–5808.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1405–1418.

Junqiu Wei, Xiaozhe Ren, Xiaoguang Li, Wenyong Huang, Yi Liao, Yasheng Wang, Jiashu Lin, Xin Jiang, Xiao Chen, and Qun Liu. 2021. NEZHA: Neural Contextualized Representation for Chinese Language Understanding. *arXiv:1909.00204 [cs]*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Dongxian Wu, Shu-tao Xia, and Yisen Wang. 2020. Adversarial Weight Perturbation Helps Robust Generalization. *arXiv:2004.05884 [cs]*.

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*, pages 9514–9528.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104.

A Appendix

A.1 More Description about Datasets

The GLUE benchmark covers various types of tasks and can be divided into three folds in general (detailed statistics are shown in Tab. 4):

- **Single sentence classification task.** CoLA (The Corpus of Linguistic Acceptability) consists of books and journals from language theory, where each sentence is marked for grammaticality. There are two labels, 0 and 1, where 0 indicates ungrammatical and 1 indicates grammatical. SST-2 (The Stanford Sentiment Treebank) contains human annotations of sentences from movie reviews and their emotions. This task is to determine the emotion of a given sentence: positive or negative.

- **Sentence pair paraphrase task.** MRPC (The Microsoft Research Paraphrase Corpus) is a paraphrase identification task, where the corpus automatically extracts the sentence pairs from online news sources, and manually annotates whether the sentence pairs are semantically equivalent. QQP (The Quora Question Pairs), a paraphrase identification task, is a corpus of question pairs from the website Quora. The target is also to determine whether a pair of sentences is semantically equivalent. STS-B (The Semantic Textual Similarity Benchmark) is a collection of sentence pairs extracted from news headlines, video titles, image titles and natural language inference data, and each pair is annotated by humans. This task can be regarded as a fine-grained five-way paraphrase identification task.

- **Natural language inference task.** MNLI (The Multi-Genre Natural Language Inference Corpus), the natural language inference task, is a collection of text implication annotation for sentence pairs through crowdsourcing. Given the premise and hypothesis, the aim is to predict whether the premise contains the assumption (entailment), contradicts the assumption (contradiction) or neither (neutral). QNLI (Question-answering NLI), a natural language inference task, is converted from another dataset (The Stanford Question Answering Dataset, SQuAD 1.0). QNLI are obtained by combining each sentence in question and context, and filtering out the sentence pair with low lexical overlap. RTE (The Recognizing Textual Entailment) is also a natural language inference task, where the samples are from news and Wikipedia. The target is to judge whether a given sentence pair is entailment or not.

A.2 More Description about Baselines

We give a brief introduction to baselines used in this paper:

- **Full Fine-Tuning (Full-FT)** is the most prevalent fine-tuning paradigm at present. When obtaining the pre-trained model on a large-scale corpus, full parameters are updated on each specific downstream task.

- **Adapter** (Houlsby et al., 2019) reduces the number of trainable parameters by updating merely the additional MLPs (with bottleneck architecture), which are inserted in each layer of model.

- **Diff-Pruning** (Guo et al., 2021) learns a perturbation variable $\Delta\theta$ for each parameter θ in PLMs, i.e., $\theta := \theta + \Delta\theta$, where the L_0-norm constraint is imposed on the variable set $\{\Delta\theta\}$ to ensure that this set is as sparse as possible.

- **Child-Tuning** (Xu et al., 2021) identifies a sub-network from the original model by calculating the parametric gradient of the Fisher Information (FIM), and then the parameters of the sub-network are updated during the fine-tuning process.

- **Compacter** (Mahabadi et al., 2021) can be regarded as a variant of Adapter, which further reduces the number of training parameters by introducing Kronecker product and shared weights.

- **Prefix-Tuning** (Li and Liang, 2021) concatenates the additional trainable parameters with the K and V in MHA. These introduced parameters are treated as continuous prompts from the perspective of prompt learning.

- **LoRA** (Hu et al., 2022), from the view of low-rank decomposition, inserts two trainable low-rank matrices into the weight matrices W^Q and W^K per layer in a parallel manner, so as to reduce the proportion of trainable parameters and avoid inference delay.

- **BitFit** (Ben-Zaken et al., 2022) is an extremely simple structured fine-tuning method. Only the bias terms of model are fine-tuned, leaving the rest frozen.

A.3 More Details about Settings & Environments

In the full-shot learning, for a range of baselines, we first refer to the hyperparameters settings in their original paper, and then perform the grid search for the batch size and learning rate. The optimal hyperparameter combinations are shown in Tab. 5. Under the few-shot learning, we directly use the size of the training set as the batch size and adopt the optimal learning rate in the full-shot learning. In addition, the sub-network structure of Child-Tuning is consistent with that in the full-shot setting to avoid too few samples to identify the sub-network effectively. Similarly, we perform the selection process of heads only once and keep them unchanged in the full/few-shot settings. Finally, we conduct the experiments on three NVIDIA GeForce RTX 3090 (24G).

A.4 More Experiments about Correlation

To verify the robustness of the head-to-head correlation, we here conduct more experiments on MRPC based on BERT$_{\text{BASE}}$. The experimental results are shown in Fig. 8, 9, 10, and 11, under the settings of **BS**, **LR**, **SL**, and **SS**, respectively.

A.5 More Detailed Algorithm Procedures

Algorithm 1 Solve for Information Richness and Correlation

Input: Training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, the PLM $f_\theta(\cdot)$ and θ represents the all parameters. The number of samples is n.

Output: The information richness I_h and correlation $r_{h,h'}$.

```
for $1 \le i \le n$ do
    Randomly sample the labelled data pair $(x_i, y_i) \sim \mathcal{D}$
    Obtain the output $O_h(x_i)$ of the $h$-th head for each layer, based on $f_\theta(x_i)$.
    Compute the $I_h(\cdot|x_i)$.
    Compute the $r(\cdot,\cdot|x_i)$.
end for
Compute the average of $I_h$ and $r_{h,h'}$.
return $I_h, r_{h,h'}, \forall\, h, h' \in \{1, 2, \cdots, H\}$.
```

Algorithm 2 Joint Optimization with PageRank

Input: The information richness I_h and correlation $r_{h,h'}$, where $h, h' \in \{1, 2, \cdots, H\}$. The upper bound of the error is ϵ, and the number of updated heads is k.

Output: The updated indicator δ_h per head.

```
Compute the initial node probability $p_h^{(0)}$ by Eq. (8) per head.
Compute the initial move probability from $h$ to $h'$ by Eq. (9).
Obtain the initial probability matrix $P^{(0)}$ and the state transition probability matrix $M$.
Compute the first time probability $P^{(1)}$ by Eq. (10).
while $\| P^{(1)} - P^{(0)} \| > \epsilon$ do
    $P^{(0)} := P^{(1)}$
    $P^{(1)} := dMP^{(1)} + \frac{1-d}{H}\mathbb{I}$
end while
Obtain the PageRank value $p_h^*$ per head.
Let $\delta_h = 1$ if $h \in Topk\{p_h^*\}$, otherwise 0.
return $\delta_h, \forall\, h \in \{1, 2, \cdots, H\}$.
```

Algorithm 3 Parameter-Efficient Fine-tuning with HiFi

Input: Training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, the PLM $f_\theta(\cdot)$ and $\theta = \{\mathcal{U}, \mathcal{V}\}$, where $\mathcal{U}/\mathcal{V}$ represents the updated/frozen weights set. $\mathcal{L}(\cdot)$ indicates the loss function and η is the learning rate.

Output: The fine-tuned heads weight set $\mathcal{U}'$.

```
// Step 1: Pre-processing
For each layer $l$, obtain the indicator $\delta_h^l \in \{0, 1\}$ w.r.t the $h$-th head weights set $W_h^l = \{W_h^Q, W_h^K, W_h^V\}$ by Alg. 1 and 2.
Let $W_h^l \in \mathcal{U}$ if $\delta_h^l = 1$, otherwise $W_h^l \in \mathcal{V}$.
Freeze the parameters of $\mathcal{V}$.
// Step 2: Fine-tuning
while not converged do
    Randomly sample the labelled data pair $(x_i, y_i) \sim \mathcal{D}$.
    Compute the loss $\mathcal{L}(\theta) = \mathcal{L}(f_\theta(x_i), y_i)$.
    for $W_h^l \in \mathcal{U}$ do
        $W_h^l := W_h^l - \eta \frac{\partial \mathcal{L}(\theta)}{\partial W_h^l}$
    end for
end while
return The fine-tuned heads weight set $\mathcal{U}'$.
```

A.6 More Analysis about Efficiency

As shown in Alg. 3, our approach consists of two phases: pre-processing and fine-tuning. In the pre-

	QNLI	SST-2	MNLI$_m$	MNLI$_{mm}$	CoLA	MRPC	STS-B	RTE	QQP
Full-shot Learning									
# Train	104,743	67,349	392,702	392,702	8,551	3,668	5,749	2,490	363,846
# Valid	5,463	872	9,815	9,796	1,043	408	1,500	277	40,430
# Test	5,463	1,821	9,832	9,847	1,063	1,725	1,379	3,000	390,965
Few-shot Learning									
# Train	16×2	16×2	16×3	16×3	16×2	16×2	-	16×2	16×2
# Valid	16×2	16×2	16×3	16×3	16×2	16×2	-	16×2	16×2
# Test	5,463	872	9,815	9,796	1,043	408	-	277	40,430

Table 4: Statistics of each dataset on GLUE in both full-shot and few-shot scenarios.

Model	QNLI	SST-2	MNLI$_m$	MNLI$_{mm}$	CoLA	MRPC	STS-B	RTE	QQP
Full-FT	32/2e-5	32/1e-5	48/2e-5	48/2e-5	32/1e-5	32/2e-5	32/2e-5	32/2e-5	32/2e-5
Diff-Pruning	32/2e-5	32/5e-5	48/1e-5	48/1e-5	32/1e-5	32/1e-5	32/1e-5	32/1e-5	32/2e-5
Child-Tuning	16/2e-5	16/4e-5	16/2e-5	16/2e-5	16/4e-5	16/4e-5	16/4e-5	16/4e-5	16/4e-5
Adapter	32/3e-4	32/1e-4	48/3e-4	48/3e-4	32/3e-4	32/2e-4	32/2e-4	32/3e-4	32/3e-4
BitFit	32/2e-4	32/4e-4	48/1e-4	48/1e-4	32/4e-4	32/2e-3	32/1e-4	32/1e-4	32/4e-4
LoRA	32/3e-4	32/2e-4	32/2e-4	32/2e-4	32/3e-4	32/1e-4	32/3e-4	32/3e-4	32/2e-4
Compactor	32/3e-3	32/3e-3	48/3e-3	48/3e-3	32/8e-4	32/2e-3	32/3e-3	32/1e-3	32/3e-3
Prefix-Tuning	32/3e-4	32/3e-4	32/5e-4	32/5e-4	32/3e-4	32/2e-4	32/3e-4	32/3e-4	32/3e-4
HiFi$_{\text{layer-wise}}$	32/2e-4	32/2e-4	32/2e-4	32/2e-4	16/1e-4	16/1e-4	16/2e-4	16/2e-4	32/1e-4
HiFi$_{\text{mid-top}}$	32/1e-4	32/2e-4	32/2e-4	32/2e-4	16/3e-4	16/2e-4	16/3e-4	16/2e-4	32/2e-4

Table 5: The settings of batch size / learning rate for varying baselines on a range of datasets.

processing stage, we need to calculate the information richness and correlation, where SVD is the most time-consuming operation in Alg. 1. Theoretically, for a $m \times n$ matrix, the time complexity of SVD is $\mathcal{O}(n^2 \times m + n \times m^2)$, but we can accelerate this process by using cuSOLVER (a GPU optimization library)[13] on CUDA. Besides, the correlation can be solved almost in linear time. For the PageRank algorithm in Alg. 2, the theoretical time complexity is $\mathcal{O}(t(\epsilon) \times n^2)$, where $t(\epsilon)$ and n refer to the number of iterations and nodes, respectively. As shown in the right of Fig. 6, $t(\epsilon) < 12$ under diverse datasets, which indicates the fast convergence of PageRank.

Overall, our pre-processing process is so efficient that the time spent on it is almost negligible compared to fine-tuning. Once the heads to be updated are determined, the fine-tuning efficiency of our method HiFi is similar to other methods, e.g., BitFit, Child-Tuning.

A.7 More Experimental Results on Commonly-used Tasks

In addition to evaluating the natural language understanding tasks in the main text, we also conduct the comparison on two widely-used benchmarks: SQuAD (Rajpurkar et al., 2016) and SWAG (Zellers et al., 2018). For each method, we train only 3 epochs on the training set and then report the result on the validation set are shown in Tab. 6.

Method	SQuAD (F1)	SWAG (Acc.)
Full-FT	90.7	85.5
Diff-Pruning	89.0	84.2
Child-Tuning	88.5	83.7
Adapter	88.0	83.8
BitFit	82.1	81.4
LoRA	89.2	83.1
Compacter	87.1	83.8
Prefix-tuning	87.7	84.3
HiFi	89.8	84.9

Table 6: The experimental results on the validation datasets of SQuAD and SWAG.

[13] https://pytorch.org/docs/stable/generated/torch.svd.html

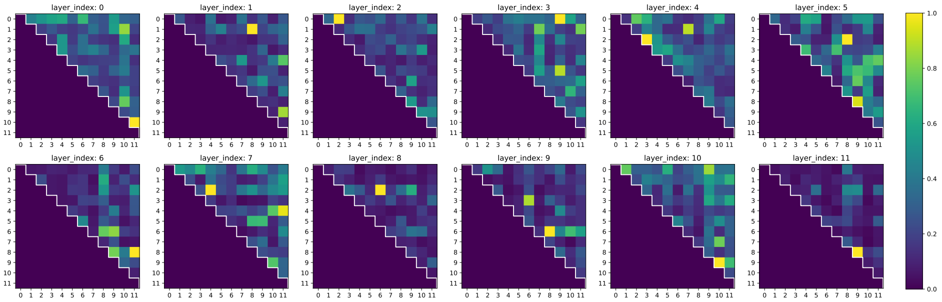

Figure 8: The effect of input batch size for the correlation between heads.

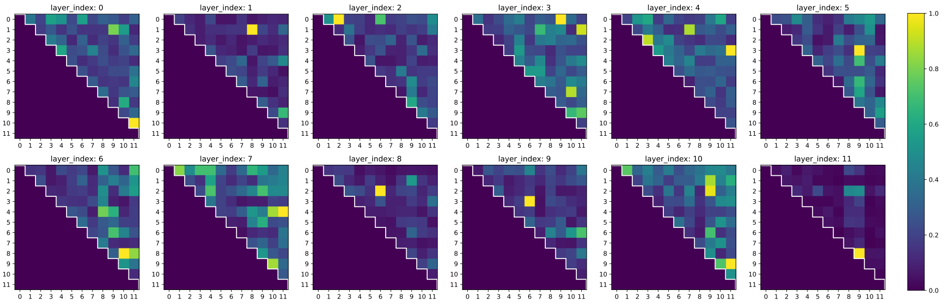

Figure 9: The effect of learning rate for the correlation between heads.

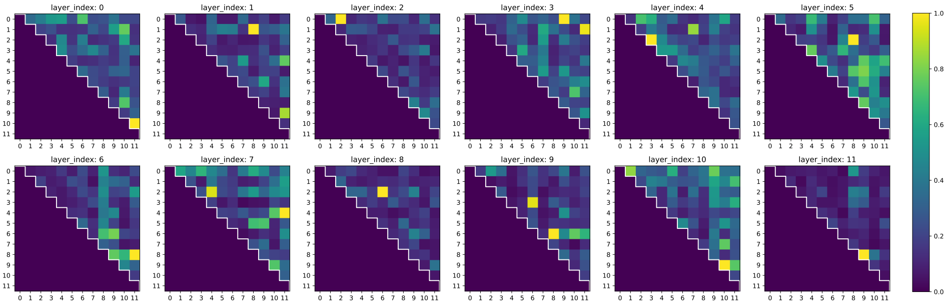

Figure 10: The effect of input sequence length for the correlation between heads.

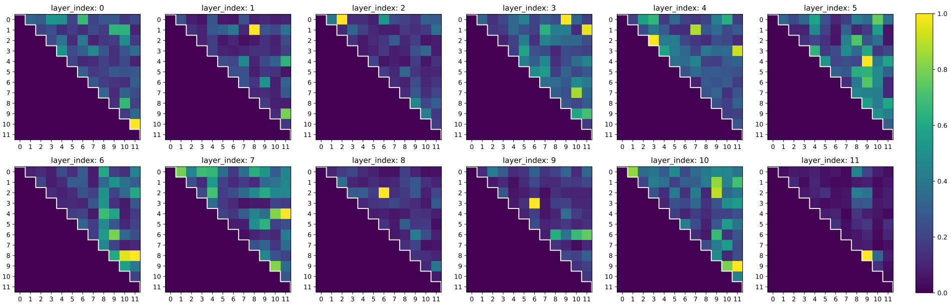

Figure 11: The effect of sample size for the correlation between heads.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7

☑ A2. Did you discuss any potential risks of your work?
Section 8

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 0, 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4

☑ B1. Did you cite the creators of artifacts you used?
Section 4

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section 8

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4, 8

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Section 8

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section A.1

C ☑ Did you run computational experiments?

Section 4, 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 5

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4

□ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
No response.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.