UNIVERSITY OF CAMBRIDGE

On the Effectiveness of Parameter-Efficient Fine-Tuning

Zihao Fu,[1] Haoran Yang,[2] Anthony Man-Cho So[2]
Wai Lam[2] Lidong Bing,[3] Nigel Collier[1]

[1]Language Technology Lab, University of Cambridge, UK
[2]The Chinese University of Hong Kong, [3]DAMO Academy, Alibaba Group

AAAI 2023

https://github.com/fuzihaofzh/AnalyzeParameterEfficientFinetune

Outline

On the Effectiveness of Parameter-Efficient Fine-Tuning

- Introduction
- Unified View of Parameter Efficient Fine-tuning
- Theoretical Analysis of the Sparse Fine-tuned Model
- Second-order Approximation Method
- Experiments
- Conclusions

Pre-Trained Model

- Fine-tuning a pre-trained model (BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), GPT (Radford et al., 2019)) has become one of the most promising techniques for NLP in recent years.
- Pre-trained models are very large.
- Very inefficient to save the fully fine-tuned parameters.
- Parameter-efficient models can solve this problem (Houlsby et al., 2019; Zaken et al., 2021; He et al., 2021).

Parameter-Efficient Fine-Tuning

- Parameter-efficient fine-tuning only tunes a small part of the original parameters and stores the tuned parameters for each task.
- Many parameter-efficient fine-tuning models have been proposed.
  - **Random Approaches** : *Random* model, *Mixout* (Lee et al., 2019)
  - **Rule-Based Approaches** : *BitFit* (Zaken et al., 2021), *MagPruning* (Han et al., 2015a,b; Lee et al., 2021), *Adapter* (Houlsby et al., 2019; Pfeiffer et al., 2020), *LoRA* (Hu et al., 2022; Karimi Mahabadi et al., 2021)
  - **Projection-Based Approaches** : *DiffPruning* (Mallya et al., 2018; Sanh et al., 2020), *ChildPruning* (Xu et al., 2021; Mostafa and Wang, 2019)

Two Questions for Parameter-Efficient Models

- This paper answers two important questions:
- Q1. why do parameter-efficient models (such as Adapters, LoRA, Bitfit, etc.) achieve promising results?
  - Our Answer: The sparsity itself improves the model stability and generalization capability.
- Q2. How to choose the tunable parameters?
  - Our Answer: Use our Second-order Approximation Method (SAM).

Main Contributions

- We propose a new categorization scheme for existing parameter-efficient methods and generalize most of these methods with a unified view called the sparse fine-tuned model.
- We conduct a theoretical analysis of the parameter-efficient models' stability and generalization.
- We propose a novel SAM model to choose the suitable parameters to optimize.
- We conduct extensive experiments to verify our theoretical analysis and the SAM model.

Outline

On the Effectiveness of Parameter-Efficient Fine-Tuning

- Introduction
- Unified View of Parameter Efficient Fine-tuning
- Theoretical Analysis of the Sparse Fine-tuned Model
- Second-order Approximation Method
- Experiments
- Conclusions

Unified View of Parameter Efficient Fine-tuning

- We first give the definition of p-sparse fine-tuned model.
- We show most of the existing parameter-efficient models are p-sparse fine-tuned models.
- We will conduct our theoretical analysis on p-sparse fine-tuned model which can naturally explain most of the existing models.

Definition 1 (p-Sparse Fine-tuned Model)

Given a pre-trained model $\mathcal{M}^0$ with parameters θ^0, if a fine-tuned model $\mathcal{M}$ with parameters θ has the same structure as $\mathcal{M}^0$ such that $\|\theta - \theta^0\|_0 \leq p \dim(\theta), p \in (0, 1)$, we say the model $\mathcal{M}$ is a p-sparse fine-tuned model with the sparsity p.

Parameter Efficient Fine-tuning Taxonomy

All the following models are p-sparse fine-tuned models.

- Random Approaches
  - **Random** model
  - **Mixout** (Lee et al., 2019)
- Rule-Based Approaches
  - **BitFit** (Zaken et al., 2021)
  - **MagPruning** (Han et al., 2015a,b; Lee et al., 2021)
  - **Adapter** (Houlsby et al., 2019; Pfeiffer et al., 2020)
  - **LoRA** (Hu et al., 2022; Karimi Mahabadi et al., 2021)
- Projection-Based Approaches
  - **DiffPruning** (Mallya et al., 2018; Sanh et al., 2020)
  - **ChildPruning** (Xu et al., 2021; Mostafa and Wang, 2019)

Random Approaches

- Random approaches randomly choose the parameters to train.
- **Random** model
  - Randomly selecting the parameters with respect to a given sparsity ratio.
- **Mixout** (Lee et al., 2019)
  - Reset a portion of the fine-tuned model's parameters to the pre-trained parameters with respect to a ratio.

Rule-Based Approaches

- The rule-based approaches directly use a pre-defined rule to choose the parameters to be tuned.
- **BitFit** (Zaken et al., 2021)
  - Only fine-tunes the bias terms.
- **MagPruning** (Han et al., 2015a,b; Lee et al., 2021)
  - Tunes the parameters with large norms.

Equivalent Model

- Some rule-based approaches add new structures to the pre-trained models.
- Introduce Equivalent Model to help analysis.
- Adapter and LoRA is p sparse fine-tuned models with respect to the equivalent model.

Definition 2 (Equivalent Model)

Given a pre-trained model $\mathcal{M}^0$ with parameters θ^0, we say that a model $\tilde{\mathcal{M}}^0$ with parameters $\tilde{\theta}^0$ is an equivalent model for model $\mathcal{M}^0$ if $\forall x, \mathcal{M}^0(x) = \tilde{\mathcal{M}}^0(x)$.

Rule-Based Approaches

- **Adapter** (Houlsby et al., 2019; Pfeiffer et al., 2020)
  - Adapter proposes to add an adapter layer inside the transformer layer. It can be viewed as fine-tuning an equivalent model shown in Fig. 1 (a).
- **LoRA** (Hu et al., 2022; Karimi Mahabadi et al., 2021)
  - LoRA proposes to add a new vector calculated by recovering a hidden vector from a lower dimension space. The model is illustrated in Fig. 1 (b).

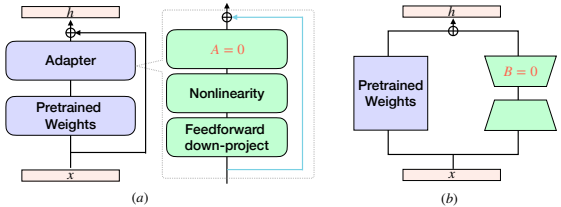

Figure 1: Equivalent model for Adapter (a) and LoRA (b).

Projection-Based Approaches

- Projection-based approaches choose and optimize the parameters alternately.
- **DiffPruning** (Mallya et al., 2018; Sanh et al., 2020)
  - DiffPruning model the parameter selection mask as a Bernoulli random variable and optimize it with the reparametrization method. It then projects the mask onto its feasible region and does the optimization alternately.
- **ChildPruning** (Mostafa and Wang, 2019; Xu et al., 2021)
  - ChildPruning iteratively trains the full model parameters and calculates the projected mask.

Projection Discontinuity Problem

- Projection-based methods suffer from the projection discontinuity problem.
- Specifically, the feasible region of the mask is non-convex. non-expansion property is not guaranteed.
- A small perturbation can lead to a totally different projection.

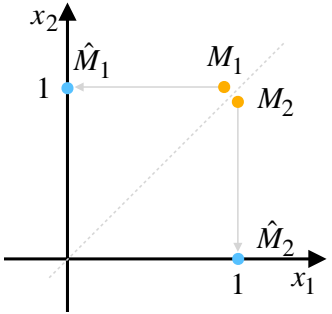

Figure 2: Projection discontinuity problem.

Outline

On the Effectiveness of Parameter-Efficient Fine-Tuning

- Introduction
- Unified View of Parameter Efficient Fine-tuning
- Theoretical Analysis of the Sparse Fine-tuned Model
- Second-order Approximation Method
- Experiments
- Conclusions

Definition of Pointwise Hypothesis Stability

We denote the original training data as $S = \{z_1, \cdots, z_n\}$ and the dataset without one sample as $S^i = S \backslash z_i = \{z_1, \cdots, z_{i-1}, z_{i+1}, \cdots, z_n\}$, where z_i is the ith training sample. We also define $i \sim U(n)$ as a sampling procedure from a uniform distribution with n samples. $\mathcal{A}(S)$ is defined as model parameters obtained by running algorithm $\mathcal{A}$ on data S.

Definition 3 (Pointwise Hypothesis Stability, (Bousquet and Elisseeff, 2002))

We say that a learning algorithm $\mathcal{A}$ has **pointwise hypothesis stability** ϵ with respect to a loss function ℓ, if

$$\mathbb{E}_{S,i\sim U(n)}[|\ell(\mathcal{A}(S^i), z_i) - \ell(\mathcal{A}(S), z_i)|] \leq \epsilon. \tag{1}$$

Stability Bound

Theorem 1 (Stability)

If the loss function ℓ is $\rho-$Lipschitz, $\mathcal{A}(S^i)$ is close to $\mathcal{A}(S)$, the Hessian matrix $\nabla^2\mathcal{L}(\mathcal{A}(S))$ at $\mathcal{A}(S)$ is positive-semidefinite with a singular value decomposition $U\,\mathrm{diag}(\Lambda)U^{-1}$, $\Lambda = \{\Lambda_1, \cdots, \Lambda_m\}$ and $\Lambda_{min} = \min\{\Lambda_1, \cdots, \Lambda_m\}$, then the expectation of the loss $\mathbb{E}_M\, L_R$ has a pointwise hypothesis stability as:

$$\mathbb{E}_{S,i\sim U(n)}[|\ell(\mathcal{A}(S^i), z_i) - \ell(\mathcal{A}(S), z_i)|] \leq \frac{2\rho^2}{(\Lambda_{min} + 2(1-p))n}. \tag{2}$$

- As the sparsity parameter p decreases, the upper bound also decreases.
- Sparse models imply better stability.
- If p is too small enough, the upper bound will not change significantly. Now, the denominator is dominated by Λ_{min}.

Generalization Bound

Theorem 2 (Generalization)

We denote the generalization error as $R(\mathcal{A}, S) = \mathbb{E}_z\, \ell(\mathcal{A}(S), z)$ *and the empirical error as* $\hat{R}(\mathcal{A}, S) = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathcal{A}(S), z)$. *Then, for some constant* C, *we have with probability* $1 - \delta$,

$$R(\mathcal{A}, S) \leq \hat{R}(\mathcal{A}, S) + \sqrt{\frac{C^2 + \frac{24C\rho^2}{\Lambda_{min}+2(1-p)}}{2n\delta}}. \tag{3}$$

- Generalization error upper bound becomes smaller as the fine-tuned parameters become sparser.
- If p is small enough, as p continue to decrease, the training error $\hat{R}(\mathcal{A}, S)$ will possibly increase when the tunable parameters are not enough to fit the data.
- Consequently, as the sparsity decreases, the generalization error will first decrease and then increase.

Outline

On the Effectiveness of Parameter-Efficient Fine-Tuning

- Introduction
- Unified View of Parameter Efficient Fine-tuning
- Theoretical Analysis of the Sparse Fine-tuned Model
- Second-order Approximation Method
- Experiments
- Conclusions

Second-order Approximation Method

- The random and rule-based approaches do not utilize the information from the task-specific data.
- The projection-based approaches suffer from the projection discontinuity problem.
- In this section we propose an approximate best method to choose the tunable parameters.

Second-order Approximation Method

- Second-order Approximation Method (SAM), namely, utilizing the data information to help decide the parameter mask while avoiding the projection discontinuity problem.
- We approximate the loss function with its second-order Taylor expansion as

$$\mathcal{L}(\theta^0 + M\Delta\theta) \approx \mathcal{L}(\theta^0) + \nabla\,\mathcal{L}(\theta^0)^{\mathrm{T}} M\Delta\theta + \frac{1}{2}(M\Delta\theta)^{\mathrm{T}} H M\Delta\theta.$$

- The Hessian matrix H is expensive to compute. We approximate the Hessian matrix as a diagonal matrix $H = \mathrm{diag}\{h_1, h_2, \cdots, h_n\}$. We also assume that H is positive semidefinite.
- Then, the optimization problem can be formulated as:

$$\begin{aligned} &\min_{\Delta\theta} \mathcal{L}(\theta^0) + \nabla\,\mathcal{L}(\theta^0)^{\mathrm{T}} M\Delta\theta + \frac{1}{2}(M\Delta\theta)^{\mathrm{T}} H M\Delta\theta \\ &s.t. \quad \|M\|_0 = \lfloor mp \rfloor; \quad M_{ij} = 0, \forall i \neq j; \quad M_{ii} \in \{0, 1\}. \end{aligned} \tag{4}$$

Optimal Parameter Mask Calculation

With the above setup, we can get the optimal parameter mask M for Problem (4) based on the following theorem:

Theorem 3

If $\hat{M}_{ii} = \mathbb{1}(\sum_{j=1}^{m} \mathbb{1}(|\frac{\nabla \mathcal{L}(\theta^0)_i^2}{h_i}| > |\frac{\nabla \mathcal{L}(\theta^0)_j^2}{h_j}|) \geq m - \lfloor mp \rfloor)$*, where* $\nabla \mathcal{L}(\theta^0)_i$ *is the* i*th element of the gradient vector* $\nabla \mathcal{L}(\theta^0)$*, then*

$$\inf_{\Delta\theta} \mathcal{L}(\theta^0 + \hat{M}\Delta\theta) \leq \inf_{\substack{\Delta\theta, \|M\|_0 = \lfloor mp \rfloor; \\ M_{ij}=0, \forall i \neq j; M_{ii} \in \{0,1\}}} \mathcal{L}(\theta^0 + M\Delta\theta). \tag{5}$$

- Selecting features according to Theorem 3 achieves the minimal value in Problem (4).
- The remaining problem is how to calculate the diagonal of the Hessian matrix.
- It is still very complex.

Second-order Approximation Method

To solve this problem, instead of minimizing the target function in Problem (4), we propose to optimize its upper bound.

$$\begin{aligned} &\min_{\Delta\theta} \mathcal{L}(\theta^0)+\nabla \mathcal{L}(\theta^0)^{\mathrm{T}} M\Delta\theta + \frac{1}{2}(M\Delta\theta)^{\mathrm{T}} DM\Delta\theta \\ &s.t. \quad \|M\|_0 = \lfloor mp \rfloor; \quad M_{ij} = 0, \forall i \neq j; \quad M_{ii} \in \{0,1\}, \end{aligned} \tag{6}$$

where $D = \text{diag}\{|\lambda_{max}|, |\lambda_{max}|, \cdots, |\lambda_{max}|\}$ and λ_{max} is the maximal eigenvalue of H.

- Second-order Approximation Method
  - (1) Get the gradient $\nabla \mathcal{L}(\theta^0)_i$ for the ith parameter θ_i.
  - (2) Calculate $|\nabla \mathcal{L}(\theta^0)_i^2|$ and take the top $\lfloor mp \rfloor$ parameters to optimize.
  - (3) Optimize selected parameters.

Outline

On the Effectiveness of Parameter-Efficient Fine-Tuning

- Introduction
- Unified View of Parameter Efficient Fine-tuning
- Theoretical Analysis of the Sparse Fine-tuned Model
- Second-order Approximation Method
- Experiments
- Conclusions

Main Experiment

	CoLA	STS-B	MRPC	RTE	CB	COPA	WSC	AVG
FullTuning	58.36±1.74	89.80±0.52	**89.55**$_{[1]}$±0.81	76.03±2.14	88.93$_{[2]}$±2.37$_{[2]}$	67.70±4.41	53.10±6.18	74.78±2.60
Random	58.35±1.05$_{[2]}$	89.81±**0.11**$_{[1]}$	88.73±0.80	72.71±3.23	**90.54**$_{[1]}$±3.39	68.80±2.64	52.88±5.97	74.55±2.46
MixOut	58.66±1.96	90.15$_{[3]}$±0.17	88.69±0.60$_{[3]}$	**77.55**$_{[1]}$±**1.64**$_{[1]}$	86.51±4.13	71.30±4.84	52.98±6.78	75.12$_{[3]}$±2.88
Bitfit	56.67±1.45	90.12±0.14$_{[3]}$	87.35±0.58$_{[2]}$	72.74±2.47	86.96±3.20	71.20±3.79	55.10±5.39	74.31±2.43
MagPruning	56.57±2.47	90.30$_{[2]}$±0.14$_{[3]}$	88.09±0.79	73.53±1.84$_{[3]}$	81.25±3.50	71.50$_{[3]}$±2.46$_{[2]}$	55.67±**2.73**$_{[1]}$	73.85±1.99$_{[2]}$
Adapter	**62.11**$_{[1]}$±1.22$_{[3]}$	90.05±0.13$_{[2]}$	89.29$_{[3]}$±0.60$_{[3]}$	76.93$_{[3]}$±2.05	87.32±4.62	69.50±2.54$_{[3]}$	57.02$_{[2]}$±5.27	76.03$_{[2]}$±2.35
LoRA	60.88$_{[3]}$±1.48	87.19±0.51	89.53$_{[2]}$±0.62	76.97$_{[2]}$±1.92	84.64±3.76	69.70±2.83	56.84$_{[3]}$±4.52	75.11±2.24$_{[3]}$
DiffPruning	58.53±1.49	89.59±0.34	78.79±6.09	69.93±7.87	86.25±2.65$_{[3]}$	72.10$_{[2]}$±2.91	53.37±3.60$_{[3]}$	72.65±3.57
ChildPruning	60.00±1.29	89.97±1.51	87.19±3.86	75.76±4.38	86.61±3.22	69.40±4.00	55.59±3.81	74.93±3.15
SAM	60.89$_{[2]}$±**0.96**$_{[1]}$	**90.59**$_{[1]}$±0.14$_{[3]}$	88.84±**0.49**$_{[1]}$	76.79±1.72$_{[2]}$	88.93$_{[2]}$±**1.75**$_{[1]}$	**74.30**$_{[1]}$±**2.45**$_{[1]}$	**59.52**$_{[1]}$±3.08$_{[2]}$	**77.12**$_{[1]}$±**1.51**$_{[1]}$

Table 1: Main experiment. We run each experiment 10 times with different random seeds and report means and standard deviations. The number in the bracket is the rank for the scores in the corresponding column.

- Most of the parameter-efficient models achieve better performance than the FullTuning model.
- Most of the parameter-efficient models are more stable than the FullTuning model.
- Random model is more stable than the FullTuning model on average.
- SAM model outperforms several baseline models in several tasks and it ranks in the top 3 of most tasks.

Projection Discontinuity Problem

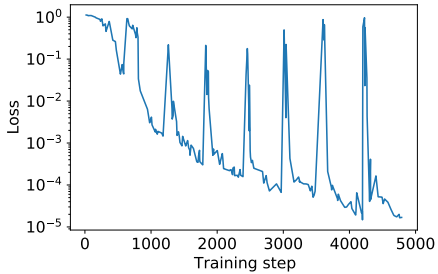

Figure 3: Projection discontinuity problem.

- We plot the training curve of the DiffPruning on the CB task which adjusts the mask every 600 training steps.
- Each time we change the mask, the training error will go back to almost the same value as its initial loss.
- This result shows that changing the mask severely affects the training procedure due to the projection discontinuity problem.

Relation between Stability and Overall Performance

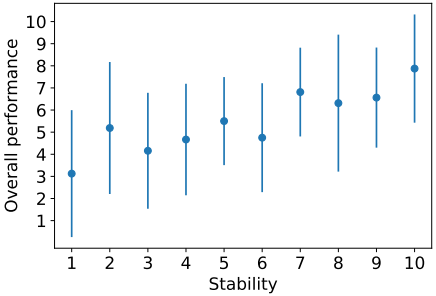

Figure 4: Relation between stability and overall Performance. We show how the stability ranks and the overall performance ranks are correlated.

- Emperically verify Theorem 2 which shows that stability implies better generalization.
- Two ranks are positively correlated indicating that stabler models usually have better generalization capability.
- Spearman's rank correlation coefficient (Spearman, 1904) for the two ranks. It can be denoted as $\rho = \frac{\text{cov}(R(S),R(V))}{\sigma_{R(S)}\sigma_{R(V)}}$, $\rho = 0.4356$ with p-value$= 0.000014 < 0.05$.

Effectiveness of sparsity

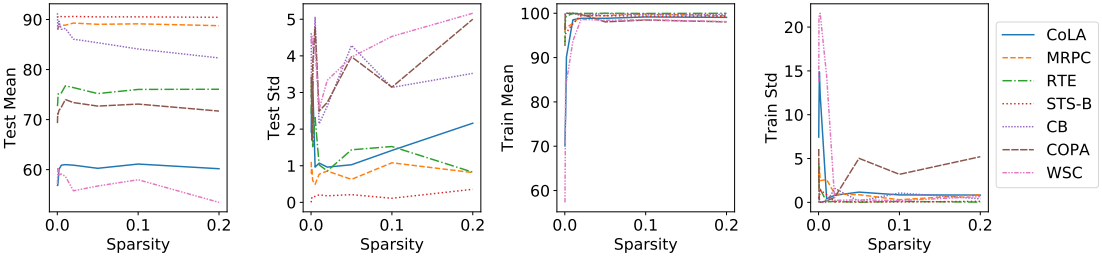

Figure 5: Effectiveness of sparsity.

- Show how the overall performance and the stability change as we change the sparsity in SAM.
- As the sparsity ratio decreases, the models become more stable with better generalization. Verifies with predictions in Theorem 1 and Theorem 2.
- If the sparsity ratio drops below a certain threshold, the models become quite unstable and the performance also sees a sharp drop. Because $\hat{R}(\mathcal{A}, S)$ becomes the dominant term and decreasing the sparsity ratio cannot further lower the bound effectively.

Data Perturbation Stability

	CoLA	STS-B	MRPC	RTE	CB	COPA	WSC	AVG
FullTuning	60.74$_{[2]}$±1.89	90.11$_{[3]}$±0.26	88.74$_{[3]}$±1.08	75.37$_{[3]}$±1.93	84.29±4.21	69.60±2.94	54.81±7.51	74.81±2.83
Random	56.00±1.84	89.79±0.20	88.57±0.72$_{[2]}$	73.00±2.01	89.29$_{[2]}$±4.92	70.30±2.69$_{[3]}$	56.87±4.29	74.83±2.38
MixOut	60.37$_{[3]}$±1.33	90.11$_{[3]}$±0.13$_{[3]}$	88.50±0.78$_{[3]}$	74.51±1.28$_{[2]}$	83.75±3.14$_{[3]}$	69.40±4.80	57.88±6.15	74.93$_{[3]}$±2.52
Bitfit	55.26±**0.78**$_{[1]}$	89.98±0.15	86.87±1.27	71.36±1.71	**91.29**$_{[1]}$±**2.27**$_{[1]}$	71.80$_{[2]}$±3.92	55.29±9.90	74.55±2.86
MagPruning	56.45±1.80	90.26$_{[2]}$±**0.11**$_{[1]}$	87.35±0.85	72.24±2.14	84.46±3.58	69.20±3.54	**59.71**$_{[1]}$±3.88$_{[2]}$	74.24±2.27$_{[2]}$
Adapter	60.05±1.88	89.92±0.19	**88.79**$_{[1]}$±0.80	74.55±1.80	86.61±4.97	68.80±**2.40**$_{[1]}$	55.63±7.53	74.91±2.79
LoRA	**61.46**$_{[1]}$±1.27$_{[3]}$	86.73±0.38	88.28±1.06	**76.46**$_{[1]}$±1.34$_{[3]}$	88.69$_{[3]}$±5.32	67.75±2.49$_{[2]}$	58.85$_{[3]}$±4.27$_{[3]}$	75.46$_{[2]}$±2.30$_{[3]}$
DiffPruning	58.36±1.45	89.52±0.27	77.46±5.31	70.76±9.01	85.18±2.65$_{[2]}$	70.40$_{[3]}$±3.07	55.38±4.30	72.44±3.72
ChildPruning	59.40±2.30	89.33±3.23	88.43±0.80	75.11±2.87	85.71±4.07	70.30±4.54	54.04±7.24	74.62±3.58
SAM	59.52±1.12$_{[2]}$	**90.45**$_{[1]}$±0.12$_{[2]}$	**88.79**$_{[1]}$±**0.69**$_{[1]}$	75.74$_{[2]}$±**1.27**$_{[1]}$	86.79±4.39	**74.00**$_{[1]}$±2.79	59.52$_{[2]}$±**3.32**$_{[1]}$	**76.40**$_{[1]}$±**1.96**$_{[1]}$

Table 2: Data perturbation stability. The setting is the same as the main experiments except that we run the experiments on different sampled datasets.

- Verify the data perturbation stability by training the model on 10 different training sets. Each of them is made by randomly removing 10% training samples from our original training set.
- Data perturbation stability performance is similar to the main experiment and our proposed SAM model still has the best data perturbation stability as well as the overall performance among all the models.

Outline

On the Effectiveness of Parameter-Efficient Fine-Tuning

- Introduction
- Unified View of Parameter Efficient Fine-tuning
- Theoretical Analysis of the Sparse Fine-tuned Model
- Second-order Approximation Method
- Experiments
- Conclusions

Conclusions

- We propose a new categorization scheme for existing parameter-efficient methods and generalize most of these methods with a unified view called the sparse fine-tuned model.
- We conduct a theoretical analysis of the parameter-efficient models' stability and generalization.
- We propose a novel SAM model to choose the suitable parameters to optimize.
- We conduct extensive experiments to verify our theoretical analysis and the SAM model.

Thank You!

https://github.com/fuzihaofzh/AnalyzeParameterEfficientFinetune

Reference I

Olivier Bousquet and André Elisseeff. 2002. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Song Han, Huizi Mao, and William J Dally. 2015a. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

Song Han, Jeff Pool, John Tran, and William Dally. 2015b. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

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*.

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

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

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2019. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*.

Jaeho Lee, Sejun Park, Sangwoo Mo, Sungsoo Ahn, and Jinwoo Shin. 2021. Layer-adaptive sparsity for the magnitude-based pruning. In *International Conference on Learning Representations*.

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

Reference II

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. 2018. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82.

Hesham Mostafa and Xin Wang. 2019. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pages 4646–4655. PMLR.

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389.

Charles Spearman. 1904. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101.

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.

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