Tuning Stable Rank Shrinkage: Aiming at the Overlooked Structural Risk in Fine-tuning

Sicong Shen[1*] **Yang Zhou**[1*] **Bingzheng Wei**[2] **Eric I-Chao Chang**[3] **Yan Xu**[1†]
[1]School of Biological Science and Medical Engineering, Beihang University
[2]Xiaomi Corporation [3]Taiwan Artificial Intelligence Foundation
{sicongshen3,xuyan04}@gmail.com zhouyangbme@buaa.edu.cn

Abstract

Existing fine-tuning methods for computer vision tasks primarily focus on re-weighting the knowledge learned from the source domain during pre-training. They aim to retain beneficial knowledge for the target domain while suppressing unfavorable knowledge. During the pre-training and fine-tuning stages, there is a notable disparity in the data scale. Consequently, it is theoretically necessary to employ a model with reduced complexity to mitigate the potential structural risk. However, our empirical investigation in this paper reveals that models fine-tuned using existing methods still manifest a high level of model complexity inherited from the pre-training stage, leading to a suboptimal stability and generalization ability. This phenomenon indicates an issue that has been overlooked in fine-tuning: Structural Risk Minimization. To address this issue caused by data scale disparity during the fine-tuning stage, we propose a simple yet effective approach called Tuning Stable Rank Shrinkage (TSRS). TSRS mitigates the structural risk during the fine-tuning stage by constraining the noise sensitivity of the target model based on stable rank theories. Through extensive experiments, we demonstrate that incorporating TSRS into fine-tuning methods leads to improved generalization ability on various tasks, regardless of whether the neural networks are based on convolution or transformer architectures. Additionally, empirical analysis reveals that TSRS enhances the robustness, convexity, and smoothness of the loss landscapes in fine-tuned models. Code is available at https://github.com/WitGotFlg/TSRS.

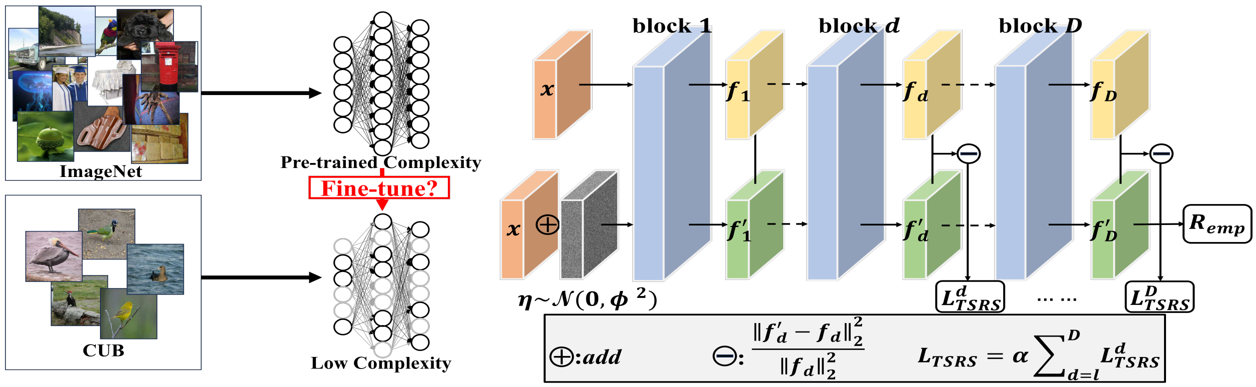
1. Introduction

Recently, deep learning has garnered remarkable performance in numerous computer vision tasks but its efficacy heavily hinges on the availability of substantial training data, a challenge that arises in various scenarios due to the exorbitant costs associated with annotation and privacy concerns [63]. Consequently, transfer learning has emerged as a proposed solution to tackle this challenge [64]. Among transfer learning methods, fine-tuning stands out as a popular approach that allows for the direct utilization of learned feature representations from a large-scale source data corpus [18, 19]. By leveraging pre-trained neural networks (NNs), fine-tuning enables satisfactory performance on target tasks, even when confronted with limited amounts of training data.

Hitherto, many methods have been proposed to improve vanilla fine-tuning by either retaining the valuable knowledge or suppressing the unfavorable one to the target task [8, 27, 33, 37, 50, 57, 60]. These methods mainly focus on re-weighting the prior knowledge of the source domain to reduce the empirical risk of the target model. However, according to the theory of structural risk minimization (SRM), a model with good generalization ability needs to exhibit both low empirical risk and low model complexity [51]. In the pre-training stage, a NN needs to have a large complexity to fit a large amount of source data, whereas in the fine-tuning stage, it necessitates a lower complexity congruous with the small amount of the target data to minimize the structural risk [4, 21, 32]. Hence, directly inheriting the pre-trained model from the source domain inevitably introduces redundant model complexity, which subsequently increases the structural risk of the model in the target domain and diminishes its generalization ability. Mainstream fine-tuning methods primarily concentrate on knowledge re-weighting and do not effectively address the issue of model complexity (see Fig. 1a). Our empirical investigations show that even if the fine-tuned model fits plausibly well in the target domain, its model complexity remains largely unchanged. Therefore, in addition to knowledge re-weighting, considering the model complexity is crucial for mitigating the structural risk of the target model during the fine-tuning process.

An intuitive approach to reducing the complexity of the fine-tuned model is to adjust its weights. However, this approach may not provide an optimal solution for transfer

*Authors contributed equally.
†Corresponding author.



(a) (b)

Figure 1. (a) Fine-tuning cannot effectively decrease the pre-trained model complexity for a smaller target dataset. (b) Illustration of Tuning Stable Rank Shrinkage (TSRS). η: noise, x: input, f_d: the feature outputted from the d^{th} block, f'_d: the feature added with noise, D: the number of network blocks, L^d_{TSRS}: the noise sensitivity loss of the d^{th} block, α: weighting hyper-parameter. L_{TSRS} is added to the objective loss to optimize the model in the direction of having lower noise sensitivity, i.e. lower stable rank and lower model complexity.

learning since determining the appropriate degree of constraint on the weights can be challenging [33]. For instance, considering L^2 normalization, inadequate weight constraints can fail to reduce the model's complexity, resulting in negative transfer [50]. Conversely, excessive weight constraints can lead to the loss of previously acquired knowledge from the source domain, leading to catastrophic forgetting [27]. Thus, there is a pressing need to investigate a more appropriate and convenient method for effectively constraining the complexity of models during the fine-tuning process.

Focusing on SRM in fine-tuning, we pointed out that attention should not only be paid to the source knowledge re-weighting but also to the complexity of the target model, which is ignored by existing fine-tuning methods. We design a simple yet effective approach called Tuning Stable Rank Shrinkage (TSRS) to reduce the structural risk of the fine-tuned model by constraining its noise sensitivity, which is theoretically the upper bound of the stable rank of NNs. The results demonstrate that TSRS yields promising outcomes and complements knowledge-reweighting-based fine-tuning methods for both convolution-based and transformer-based neural networks. Additionally, empirical analysis reveals that TSRS enhances the robustness, convexity, and smoothness of the loss landscapes in fine-tuned models. Our contributions are summarized as follows:

- We point out the importance of considering the structural risk of the target model in the fine-tuning process, which is often overlooked in current fine-tuning methods.
- We design Tuning Stable Rank Shrinkage (TSRS), a straightforward yet effective approach for reducing the structural risk of the fine-tuned model by constraining its noise sensitivity.
- Extensive experiments and analyses demonstrate the effectiveness and generality of TSRS.

2. Related Work

In computer vision, Fine-tuning [11, 18, 19] is a widely used transfer learning [64] approach to help a NN converge and perform better on the target tasks [7, 14]. During the fine-tuning process, many researchers have made improvements to the baseline fine-tuning to reduce generalization error on the target domain [1]. Recent works on fine-tuning mainly focus on how to better re-weight different parts of knowledge of pre-trained models, i.e. how to retain the parts conducive to the target domain [27, 33, 37, 57, 60] or suppress the detrimental ones [8, 50]. Approaches such as DELTA [33], Co-Tuning [60], UOTS [37], and BSS [8] have been proposed to achieve these goals by incorporating feature map constraints, task layer preservation, optimal transport selection, and spectral shrinkage strategies, respectively. These approaches that involve knowledge re-weighting fundamentally aim to reduce empirical risk [52], which is a focus of research in the field of deep learning for enhancing model performance [12, 39, 62]. Despite the success of existing methods in reducing empirical risk through re-weighting pre-trained knowledge, we find that structural risk minimization (SRM) in transfer learning is also critical, which has been ignored by these methods [51]. In fact, through experiments, we show that the complexity of the fine-tuned models, i.e. structural risk, using these methods is still high, revealing that ignoring SRM has resulted in a suboptimal generalization ability in fine-tuning.

The SRM principle selects the model by balancing the model complexity against its success at fitting the training data [1, 51]. SRM has been discussed in pruning [16, 35, 36], knowledge distillation [20]. These applications involve adjusting the network structure after training, and their main goal is to achieve efficient inference rather than better generalization ability. Nevertheless, SRM has the potential to improve the model's generalization ability [3]. To explore SRM in NLP fine-tuning, Hua *et al.* [23] constrain the noise perturbation of the language model to stabilize the BERT [25] fine-tuning. Stable rank [47] is used in several areas of mathematics such as algebraic K-theory, operator algebras, and algebraic geometry [53]. Recently, Sanyal *et al.* [48] first propose to use stable rank to constrain structural risk, demonstrating that constraining the stable rank can improve the model's generalization ability. But their method only constrained the structural risk of the final classification layer through an iterative mode. In fact, there is a strong correlation between the noise sensitivity and the stable rank of a NN [3]. Inspired by the aforementioned work, we propose TSRS, which uses noise sensitivity to estimate the stable rank of the entire NN in order to reduce the structural risk of the fine-tuned model that other fine-tune methods overlook.

3. Neglection of Structural Risk Minimization (SRM) in Fine-tuning

To facilitate later analysis, let us first formulate the concept of model structural risk during fine-tuning. M, w_t denote the model and its target weights. $D_s = \{(x_s^i, y_s^i)\}_{i=1}^{m_s}$ and $D_t = \{(x_t^i, y_t^i)\}_{i=1}^{m_t}$ denote the source domain data and the target domain data. Denote the **empirical risk** and the penalty loss on model complexity by R_{emp}, L_{srm}, the **structural risk** is formularized as:

$$R_{srm} = R_{emp} + L_{srm}. \tag{1}$$

3.1. Revisiting existing fine-tuning methods

In fine-tuning, the empirical risk can be divided into two parts: one is the empirical risk of newly learned knowledge in the target domain D_t, denoted as L_{emp}, and the other is the empirical risk of inherited knowledge from the source domain, denoted as L_{temp}, which is formularized as:

$$R_{emp} = L_{emp}\left(x_t, y_t, w_t\right) + L_{temp}\left(x_t, y_t, w_t, w_s\right), \tag{2}$$

$$L_{emp} = \frac{1}{m_t}\sum_{i=1}^{m_t} L\left(M\left(x_t^i, w_t\right), y_t^i\right), \tag{3}$$

where $L(\cdot)$ denotes the loss function.

Existing fine-tuning methods focus on two aspects for improvement. Firstly, some of them aim to prevent catastrophic forgetting [8, 27] by retaining beneficial knowledge obtained from the source domain. DELTA forces the alignment of the feature maps of the fine-tuned model and the pre-trained model. Secondly, the other methods aim to avoid negative knowledge transfer [50] by suppressing unfavorable knowledge obtained by the pre-trained model. For instance, BSS suppresses small feature components during fine-tuning. Both aspects form a L_{temp} to better optimize R_{emp} and can be unified under the concept of knowledge re-weighting. Detailed formulations are given in the supplement.

3.2. Estimation of sRM in fine-tuning

The aforementioned methods mainly focus on re-weighting knowledge of the source domain to reduce the empirical risk R_{emp} associated with the target model. However, the model selected merely by empirical risk minimization may be suboptimal. According to the SRM principle, complex tasks require large models with large complexity, while simple tasks require models with low complexity [51]. In the pre-training stage, a NN needs to have a large model complexity to accommodate the extensive data samples. However, during the fine-tuning stage, where the target data is limited, it is theoretically essential to employ a model with reduced complexity to mitigate the potential structural risk. Regrettably, existing fine-tuning methods do not explicitly consider the SRM principle. Empirical analysis shows that the model complexity of these existing methods remains almost unchanged during the fine-tuning process.

3.2.1 Stable rank and noise sensibility

Proper model complexity estimation is essential. Recent research has proposed norm-based methods [34] and sensitivity-based methods [40, 43] for complexity estimation. However, these methods are limited in their ability to estimate the complexity of large models such as ResNet [17] and ViT [13], as they are primarily designed for simpler neural networks with fewer layers [22]. For NNs with fixed architectures, their complexity is mainly reflected in the coefficient levels of their weight matrices, making the rank of weight matrices a potential tool for model complexity estimation. Unfortunately, the exact rank is unstable under small perturbations, and commonly used network structures, such as residual connections, have permutation symmetry-breaking capacity, which leads to a high exact rank even for sparse weight matrices [43]. Therefore, the exact rank is not suitable for estimating the complexity of the model. To address this issue, we introduce stable rank here, which is defined as the square of the ratio between the Frobenius norm and the spectral norm of a matrix $\mathbf{W}$:

$$\text{srank}(\mathbf{W}) = \frac{\|\mathbf{W}\|_F^2}{\|\mathbf{W}\|_2^2} = \frac{\sum_{i=1}^{k}\sigma_i^2(\mathbf{W})}{\sigma_1^2(\mathbf{W})}. \tag{4}$$

Stable rank is a stable relaxation of rank [47]. Specifically, stable rank is smaller than the exact rank and less sensitive to small perturbations. Furthermore, its value is highly positively correlated with the sparsity of the matrix, even when the exact rank of the model is high.

However, directly computing the stable rank of the weight matrix of an entire NN is difficult. Therefore, we propose to use noise sensitivity as an approximation of the stable rank for estimating the model complexity. As shown by Arora *et al.* [3], there is a strong correlation between the noise sensitivity and the stable rank of a model. Concretely, a low noise sensitivity corresponds to a weight matrix having some large singular values, i.e. having a lower stable rank. Given a Gaussian noise η with noise intensity ϕ, the noise sensitivity of a model M with weight w is defined as:

$$\psi(M,x) = \mathop{\mathbb{E}}_{\eta\in\mathcal{N}}\left[\frac{\|M(x+\eta\|x\|,w)-M(x,w)\|^2}{\|M(x,w)\|^2}\right]. \tag{5}$$

Given $\eta \in \mathcal{N}(0,\phi^2)$, we have the following guarantee: $\mathrm{srank}(w) \le \mathcal{O}\left(\psi(M,x)_w\right)$. Therefore, the noise sensitivity is the upper bound of stable rank [3], i.e. the model complexity can be estimated by the noise sensitivity. Detailed proof is given in the supplementary. Based on the aforementioned reasons, we use noise sensitivity to estimate the upper bound of the stable rank and further evaluate its model complexity.

3.2.2 Empirical analysis

The above theories indicate that we can use noise sensitivity to analyze the structural risk of a NN. In Fig. 2, we display the noise sensitivity curves of ResNet50 on the CUB-200-2011 (CUB) [55] dataset before and after fine-tuning. The Figure exhibits two phenomena: (1) Before fine-tuning, the noise in the input is amplified with loading pre-trained weights (line "WP") compared with random weight initialization (line "WOP"), which reflects the high noise sensitivity of the pre-trained model in the target domain. This phenomenon proves the significant redundancy in the complexity of pre-trained models. (2) Even after fine-tuning, the problem of high noise sensitivity of loading pre-trained weights has not been suppressed (line "WP+"). This indicates that the model cannot reduce noise sensitivity solely through fine-tuning. We further found that other improved fine-tuning methods cannot reduce noise sensitivity either (please refer to Sec. 5.2 and Fig. 3 for detailed results).

4. Method

The above analysis verifies the excess structural risk in the fine-tuning scenario. Moreover, relying solely on the model's optimization during fine-tuning cannot effectively constrain the complexity of the fine-tuned model. To address this issue, a possible solution is to use stable rank as the constraint to reduce the mode complexity during fine-tuning. The generalization boundary of NNs is related to the stable rank of the weight matrices as follows:

$$\mathcal{O}\left(\sqrt{\prod_{i=1}^{D}\|\mathbf{W}_i\|_2^2\sum_{i=1}^{D}\mathrm{srank}\left(\mathbf{W}_i\right)}\right), \tag{6}$$

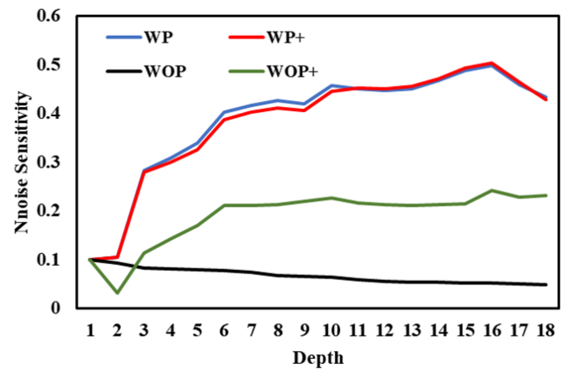


Figure 2. The noise sensitivity of the ResNet50 model fine-tuned on the CUB dataset. X-axis: the index of bottlenecks. Y-axis: noise sensitivity ψ. "WP": loading pre-trained weights. "WOP": random weight initialization. "+": after fine-tuning. Noise η with ϕ being 10% of $\|x\|$ is added to x in the first layer. The noise sensitivity of each bottleneck is calculated using Eq. (9).

where D refers to the number of weight matrices [41]. In other words, reducing the stable rank is equivalent to constraining the generalization boundary, which can improve the generalization ability of NNs. Considering previous theoretical analysis, reducing the stable rank via noise sensitivity constraint may reduce model complexity effectively.

Tuning Stable Rank Shrinkage (TSRS). We propose a new regularization method called Tuning Stable Rank Shrinkage (TSRS), which limits the upper bound of stable rank by reducing noise sensitivity and thus suppressing the redundant model complexity in fine-tuning. As illustrated in Fig. 1b, TSRS has the following steps. (1) In each forward propagation, sample a noise $\eta \in \mathcal{N}(0,\phi^2)$ and add it to the input x. (2) Calculate the sum of the noise sensitivity loss L^d_{TSRS} of each block d starting from the l^{th} block, as shown in Eq. (7), where α refers to a weighting hyper-parameter, f_d refers to the feature outputted from the d^{th} block, f'_d refers to the feature added with noise, and D refers to the total number of blocks. In this paper, we use the term "block" to denote the fundamental repeating unit in neural networks. Specifically, in the ResNet architecture, a block corresponds to a stage stacked by several bottleneck architectures, while in the case of ViT, a block refers to a transformer unit. (3) Add L_{TSRS} to the optimization objective and optimize the model in the direction of having lower noise sensitivity, i.e. lower stable rank and lower model complexity.

$$L_{TSRS} = \alpha\sum_{d=l}^{D}\frac{\|f'_d - f_d\|_2^2}{\|f_d\|_2^2} \ge srank(\mathbf{W}). \tag{7}$$

Models with TSRS. Many current fine-tuning methods focus on minimizing empirical risk R_{emp} of transfer learning, without considering the principle of structural risk minimization (SRM). TSRS is a new regularization method proposed from the perspective of reducing structural risk, which limits the upper bound of stable rank by decreasing noise sensitivity. TSRS is a straightforward and flexible method that can be embedded into existing fine-tuning methods. The incorporation of TSRS into the objective of SRM can be expressed as:

$$R_{srm} = R_{emp} + L_{srm} = R_{emp} + L_{TSRS}. \tag{8}$$

5. Experiments

We conducted comprehensive experiments to evaluate the effectiveness of our proposed TSRS. We integrated TSRS into several representative fine-tuning methods, namely BSS, DELTA, Co-Tuning, and UOTS. The performance of these methods was evaluated on various visual recognition benchmarks. To assess the universality of our approach, we tested the fine-tuning methods on both convolutional neural networks (CNNs), which are commonly used in prior work, and transformers that heavily rely on large-scale pre-trained datasets. Additionally, we conducted experiments to demonstrate the impact of TSRS on the robustness, convexity, and smoothness of loss landscapes in fine-tuned models. These experiments provided empirical evidence of the advantages of incorporating TSRS into the fine-tuning process.

5.1. Setup

In our experiments, we followed the established procedures outlined in the TLlib [1] as presented by Jiang *et al.* [24]. We conducted experiments using both CNN and Transformer models pre-trained on the ImageNet dataset. Specifically, we used ResNet50 for the CNN architecture and ViT-B for the Transformer architecture. During the fine-tuning process, we trained the final task layer from scratch, with its learning rate set to 10 times that of the pre-trained layers, as suggested by Yosinski *et al.* [59]. We used the CUB [55], Stanford Cars (Cars) [28], and FGVC Aircraft (Aircraft) [38] datasets for experiments, which were widely studied in transfer learning [60]. Experiments were conducted with different sampling rates of training examples per category: 15%, 30%, 50%, and 100%, following Chen *et al.* [8]. All models were optimized by SGD with 0.9 momentum and 0.0005 weight decay for 20 epochs. The learning rates were reduced by a factor of 1/10 in the 12^{th} epoch. Please refer to the supplement for more detailed implementation and hyper-parameters.

5.2. Results and analysis

Tab. 1 shows the experimental results for different model architectures, datasets, and sampling rates. In all cases, the methods embedded with TSRS exhibited improved performance, highlighting the synergistic benefits of employing TSRS. Notably, on the Cars and Aircraft datasets, TSRS yielded even greater improvements, particularly in scenarios where data was limited. This observation suggests that as the availability of data decreases, the redundancy in model complexity becomes more prominent, and our proposed method effectively addresses this issue, leading to enhanced optimization. Despite necessitating an extra forward pass for training, increasing the training time of one epoch from 145s to 187s on CUB, TSRS's inference time remains unchanged. This trade-off is reasonable given the benefits it offers.

Fig. 3 illustrates the noise sensitivity curves with different fine-tuning methods on the CUB dataset. The noise sensitivity of models fine-tuned using the original methods (red solid line) shows nearly no decrease compared to the pre-fine-tuning stage (blue solid line). However, embedding TSRS (red dashed line) significantly reduces the noise sensitivity. Notably, the noise sensitivity of DELTA (Fig. 3c) was even higher than before fine-tuning, which can be attributed to its retention of the behavioral characteristics of the source network. To validate this hypothesis, we calculated the noise sensitivity using ImageNet-1k as the test input x on the model fine-tuned by the original DELTA (blue dashed line). It was observed that the noise sensitivity tested on ImageNet-1k closely matched the noise sensitivity curve of the model fine-tuned by the original DELTA (red solid line), thus confirming the hypothesis. Similarly, UOTS (Fig. 3d) also exhibited a similar phenomenon as it uses a large amount of source data in fine-tuning.

5.3. Comparison with other regularization methods

Currently, there are several regularization methods designed for general circumstances to constrain model complexity during the training process. However, these methods fail to address the excess structural risk that emerges in fine-tuning scenarios, leading to negative transfer. To demonstrate this, we conducted comparative experiments on the Cars dataset using 15% of the training data, directly comparing TSRS with some representative regularization methods. This experiment employs an ImageNet pre-trained Resnet50 model and uses BSS without any regularization as the baseline. Please note that L2 regularization is the default regularization method used in BSS. The results are listed in Tab. 2, which shows that TSRS outperforms other regularization methods. This superiority is because other methods, such as L2 regularization, might have already been used during pre-training, or they solely focus on model sparsity (such as ART) without considering the differences between source and target domains in transfer learning.

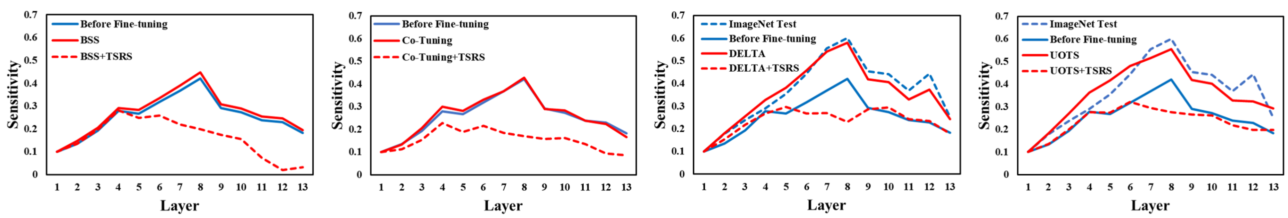
5.4. Ablation studies

Hyper-parameters. We conducted an investigation into the hyper-parameters of our method on the CUB dataset. Experiments are conducted with 15% sampling rates of training examples per category. As shown in Fig. 4a, the perfor-

[1] https://github.com/thuml/Transfer-Learning-Library

Table 1. Top-1 accuracy (%) of several methods with or without TSRS, including vanilla fine-tuning (Baseline), DELTA (2019) [33], BSS (2019) [8], Co-Tuning (2020) [60], and UOTS (2022) [37], fine-tuning on the CUB-200-2011 (CUB) [55], Stanford Cars (Cars) [28], and FGVC Aircraft (Aircraft) [38]. "Arch." refers to "Architecture". The bold marks the highest accuracy under the corresponding dataset.

Arch.	Method	CUB				Cars				Aircraft			
		15%	30%	50%	100%	15%	30%	50%	100%	15%	30%	50%	100%
ResNet50	Baseline	45.25$_{\pm 0.12}$	59.68$_{\pm 0.21}$	70.12$_{\pm 0.29}$	78.01$_{\pm 0.16}$	36.77$_{\pm 0.12}$	60.63$_{\pm 0.18}$	75.10$_{\pm 0.21}$	87.20$_{\pm 0.19}$	39.57$_{\pm 0.20}$	57.46$_{\pm 0.12}$	67.93$_{\pm 0.28}$	81.13$_{\pm 0.21}$
	Baseline+TSRS	52.76$_{\pm 0.21}$	65.22$_{\pm 0.18}$	75.16$_{\pm 0.07}$	81.96$_{\pm 0.07}$	44.15$_{\pm 0.13}$	68.36$_{\pm 0.39}$	79.68$_{\pm 0.10}$	88.81$_{\pm 0.23}$	44.40$_{\pm 0.32}$	61.18$_{\pm 0.16}$	71.83$_{\pm 0.32}$	82.33$_{\pm 0.17}$
	BSS	53.49$_{\pm 0.19}$	66.69$_{\pm 0.09}$	75.03$_{\pm 0.07}$	81.77$_{\pm 0.09}$	44.00$_{\pm 0.32}$	67.44$_{\pm 0.04}$	79.18$_{\pm 0.14}$	88.22$_{\pm 0.11}$	43.59$_{\pm 0.04}$	60.28$_{\pm 0.12}$	69.61$_{\pm 0.09}$	82.24$_{\pm 0.16}$
	BSS+TSRS	54.18$_{\pm 0.15}$	68.10$_{\pm 0.39}$	76.68$_{\pm 0.14}$	82.72$_{\pm 0.12}$	47.22$_{\pm 0.23}$	70.59$_{\pm 0.37}$	80.86$_{\pm 0.17}$	88.88$_{\pm 0.03}$	46.98$_{\pm 0.16}$	63.61$_{\pm 0.21}$	72.43$_{\pm 0.13}$	83.02$_{\pm 0.05}$
	Co-Tuning	57.78$_{\pm 0.06}$	70.50$_{\pm 0.10}$	77.30$_{\pm 0.07}$	82.76$_{\pm 0.15}$	47.94$_{\pm 0.02}$	70.75$_{\pm 0.07}$	81.46$_{\pm 0.12}$	89.03$_{\pm 0.07}$	45.00$_{\pm 0.04}$	60.49$_{\pm 0.15}$	70.78$_{\pm 0.15}$	82.27$_{\pm 0.10}$
	Co-Tuning+TSRS	**58.20$_{\pm 0.09}$**	**70.68$_{\pm 0.25}$**	**77.84$_{\pm 0.26}$**	**83.12$_{\pm 0.09}$**	**51.98$_{\pm 0.11}$**	**73.65$_{\pm 0.29}$**	**82.71$_{\pm 0.17}$**	89.38$_{\pm 0.11}$	**47.62$_{\pm 0.12}$**	**64.75$_{\pm 0.53}$**	**73.75$_{\pm 0.06}$**	**84.79$_{\pm 0.39}$**
	DELTA	54.95$_{\pm 0.01}$	67.40$_{\pm 0.07}$	76.03$_{\pm 0.05}$	82.36$_{\pm 0.13}$	44.66$_{\pm 0.24}$	68.83$_{\pm 0.20}$	79.69$_{\pm 0.12}$	88.21$_{\pm 0.06}$	44.97$_{\pm 0.18}$	61.27$_{\pm 0.02}$	71.53$_{\pm 0.04}$	82.72$_{\pm 0.21}$
	DELTA+TSRS	55.04$_{\pm 0.22}$	67.71$_{\pm 0.05}$	76.53$_{\pm 0.15}$	82.64$_{\pm 0.07}$	47.52$_{\pm 0.14}$	70.78$_{\pm 0.12}$	81.52$_{\pm 0.14}$	88.87$_{\pm 0.25}$	47.32$_{\pm 0.24}$	64.21$_{\pm 0.15}$	73.27$_{\pm 0.15}$	83.74$_{\pm 0.19}$
	UOT	55.17$_{\pm 0.14}$	66.90$_{\pm 0.21}$	74.99$_{\pm 0.12}$	81.16$_{\pm 0.27}$	42.85$_{\pm 0.54}$	67.26$_{\pm 0.46}$	79.89$_{\pm 0.33}$	89.67$_{\pm 0.19}$	39.29$_{\pm 0.39}$	56.07$_{\pm 0.32}$	67.32$_{\pm 0.40}$	80.77$_{\pm 0.18}$
	UOT+TSRS	56.18$_{\pm 0.46}$	67.94$_{\pm 0.12}$	75.78$_{\pm 0.20}$	82.20$_{\pm 0.22}$	46.09$_{\pm 0.32}$	69.80$_{\pm 0.67}$	80.81$_{\pm 0.43}$	**90.17$_{\pm 0.21}$**	41.22$_{\pm 0.42}$	58.63$_{\pm 0.37}$	69.75$_{\pm 0.13}$	82.60$_{\pm 0.26}$
ViT-B	Baseline	64.96$_{\pm 0.10}$	77.13$_{\pm 0.05}$	81.02$_{\pm 0.04}$	85.38$_{\pm 0.07}$	43.50$_{\pm 0.12}$	66.22$_{\pm 0.05}$	77.86$_{\pm 0.06}$	88.06$_{\pm 0.03}$	38.67$_{\pm 0.08}$	56.32$_{\pm 0.01}$	65.86$_{\pm 0.06}$	77.47$_{\pm 0.06}$
	Baseline+TSRS	70.76$_{\pm 0.04}$	78.44$_{\pm 0.11}$	82.02$_{\pm 0.13}$	86.40$_{\pm 0.10}$	43.88$_{\pm 0.06}$	67.52$_{\pm 0.05}$	79.01$_{\pm 0.05}$	88.10$_{\pm 0.08}$	39.24$_{\pm 0.03}$	56.74$_{\pm 0.05}$	66.94$_{\pm 0.09}$	77.80$_{\pm 0.12}$
	BSS	69.80$_{\pm 0.24}$	78.91$_{\pm 0.34}$	82.88$_{\pm 0.18}$	86.80$_{\pm 0.07}$	47.83$_{\pm 0.07}$	72.02$_{\pm 0.07}$	81.30$_{\pm 0.10}$	88.19$_{\pm 0.06}$	41.91$_{\pm 0.25}$	59.05$_{\pm 0.10}$	69.64$_{\pm 0.64}$	79.72$_{\pm 0.32}$
	BSS+TSRS	72.21$_{\pm 0.47}$	79.32$_{\pm 0.19}$	83.05$_{\pm 0.12}$	87.28$_{\pm 0.04}$	**49.72$_{\pm 0.42}$**	**72.20$_{\pm 0.29}$**	**81.31$_{\pm 0.09}$**	88.45$_{\pm 0.07}$	**43.95$_{\pm 0.13}$**	**61.15$_{\pm 0.21}$**	**70.48$_{\pm 0.13}$**	**79.84$_{\pm 0.20}$**
	Co-Tuning	76.03$_{\pm 0.18}$	82.03$_{\pm 0.03}$	84.93$_{\pm 0.10}$	87.09$_{\pm 0.02}$	44.71$_{\pm 0.07}$	69.16$_{\pm 0.04}$	80.14$_{\pm 0.01}$	88.53$_{\pm 0.19}$	39.48$_{\pm 0.15}$	57.10$_{\pm 0.10}$	66.82$_{\pm 0.07}$	77.50$_{\pm 0.16}$
	Co-Tuning+TSRS	76.51$_{\pm 0.15}$	82.38$_{\pm 0.06}$	85.23$_{\pm 0.11}$	87.68$_{\pm 0.14}$	46.64$_{\pm 0.20}$	69.89$_{\pm 0.12}$	80.14$_{\pm 0.17}$	88.72$_{\pm 0.09}$	41.82$_{\pm 0.39}$	58.21$_{\pm 0.07}$	68.05$_{\pm 0.15}$	78.04$_{\pm 0.09}$
	DELTA	76.63$_{\pm 0.04}$	82.21$_{\pm 0.65}$	85.52$_{\pm 0.09}$	88.25$_{\pm 0.12}$	47.66$_{\pm 0.07}$	68.39$_{\pm 0.12}$	79.14$_{\pm 0.17}$	89.82$_{\pm 0.08}$	40.35$_{\pm 0.11}$	56.71$_{\pm 0.16}$	67.69$_{\pm 0.06}$	78.70$_{\pm 0.21}$
	DELTA+TSRS	**76.94$_{\pm 0.03}$**	**83.12$_{\pm 0.22}$**	**86.18$_{\pm 0.07}$**	88.45$_{\pm 0.10}$	47.77$_{\pm 0.24}$	69.64$_{\pm 0.03}$	79.19$_{\pm 0.19}$	**89.96$_{\pm 0.14}$**	41.31$_{\pm 0.15}$	57.88$_{\pm 0.20}$	68.20$_{\pm 0.08}$	79.36$_{\pm 0.35}$
	UOT	73.92$_{\pm 0.61}$	81.60$_{\pm 0.22}$	85.32$_{\pm 0.47}$	88.35$_{\pm 0.35}$	46.20$_{\pm 0.37}$	67.83$_{\pm 0.71}$	78.03$_{\pm 0.61}$	88.48$_{\pm 0.39}$	40.59$_{\pm 0.30}$	56.77$_{\pm 0.47}$	67.18$_{\pm 0.24}$	77.71$_{\pm 0.52}$
	UOT+TSRS	74.07$_{\pm 0.56}$	82.12$_{\pm 0.37}$	85.66$_{\pm 0.48}$	**88.54$_{\pm 0.15}$**	46.55$_{\pm 0.23}$	68.26$_{\pm 0.14}$	78.74$_{\pm 0.31}$	88.58$_{\pm 0.27}$	41.22$_{\pm 0.11}$	56.92$_{\pm 0.11}$	67.39$_{\pm 0.13}$	78.13$_{\pm 0.44}$



(a) BSS (b) Co-Tuning (c) DELTA (d) UOTS

Figure 3. The noise sensitivity before and after fine-tuning ViT-B with different methods on the CUB dataset with ϕ being 10% of $\|x\|$. X-axis: the index of transformer blocks. The Blue dashed line represents the noise sensitivity tested on the ImageNet-1k, indicating the noise sensitivity on the source domain. Other curves are tested on the CUB dataset.

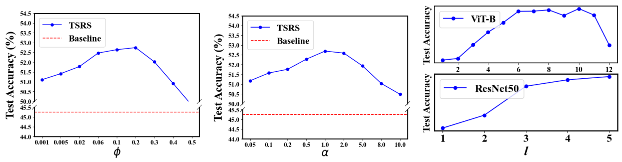
Table 2. Comparison for fine-tuning Resnet50 with other regularization methods on the Cars dataset using 15% training data. Baseline: BSS without any regularization (1^{st} column).

Regularization	-	L1 [54]	L2 [42]	SRN [48]	Dynamic [56]	ART [15]	TSRS
Accuracy(%)	43.17	42.96	44.00	44.20	43.30	43.83	**46.30**

mance of the model gradually improved as the noise intensity increased, indicating the effectiveness of regularization. However, when the noise intensity became too high, the excessive disruption of information led to a decrease in model performance. A similar trend was observed for the weighting hyper-parameter α (Fig. 4b).

Fig. 4c shows the experiment of changing l, the starting block of the applied constraint, after which all blocks of the model are constrained. The performance of the model initially improved and then stabilized as the constraint starting block shifted from shallow to deep. This is because adding noise to the input is essentially an input divergence. Then, in the blocks with L_{TSRS}, all outputs of the diverged inputs are constrained to the same point, which is substantially a compression and clustering of the data space. Please refer to the supplement for a detailed description. The scatter plots in Fig. 5 demonstrate this point well. Specifically, at the shallow blocks, due to insufficient feature extraction, the distance between intra-class features may be larger than that of inter-class features. This causes the L_{TSRS} constraint added in the shallow blocks to force the features of different classes' samples to cluster together, ultimately impairing the model's performance.

Model Size. We used the pre-trained ResNet family as



(a) Effect of ϕ (b) Effect of α (c) Effect of l

Figure 4. Effect of noise intensity (ϕ), weighting hyper-parameter(α) and the starting block (l) on the validation performance for fine-tuned ResNet50, and ViT-B on the CUB dataset.

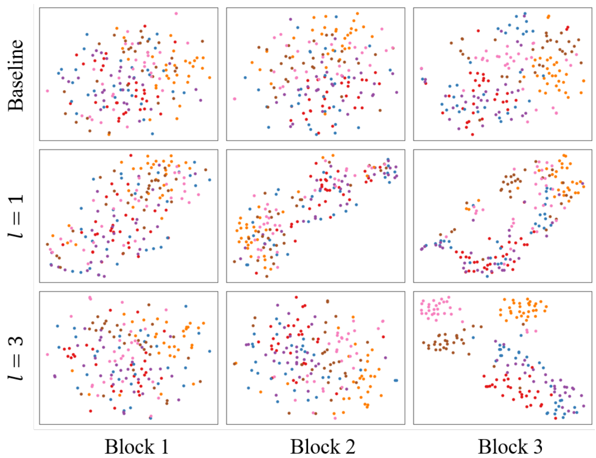


Figure 5. The t-SNE visualizations on the output of each block of ResNet50 fine-tuned with varied hyper-parameter l, on the CUB dataset [29]. Baseline refers to standard fine-tuning. The colored dots represent data points of different classes. Small l hampers the model's ability to distinguish features at low levels due to insufficient feature extraction.

an example to explore the fine-tuning performance gains of TSRS on models with varying numbers of parameters using BSS as a baseline on the Cars dataset using 15% training data, with the results presented in Tab. 3. It is observed that the larger a model is, the more significant the improvement brought by TSRS. This effect can be attributed to the increase in the number of parameters, which in turn elevates the structural risk during fine-tuning. TSRS effectively reduces this structural risk, leading to an increase in corresponding benefits. Importantly, in cases where TSRS is not utilized, the largest model, ResNet152, inherently exhibits superior fine-tuning performance. Yet, the integration of TSRS in these scenarios results in even higher benefits.

5.5. More scenarios

We validated the efficacy of TSRS across a broader range of computer vision tasks in the fine-tuning scenario.

Segmentation. We incorporated TSRS into the TransUNet [6] model, using vanilla fine-tuning as a baseline, and evaluated the performance on three datasets: Synapse multi-organ[2], ISIC2018 [10], and MRSpineSeg[3], in terms of Dice Similarity Coefficient (DSC) and Hausdorff Distance (HD). Tab. 4 displays the average segmentation results for the three datasets. It is observed that TSRS also promotes the performance of fine-tuning in image segmentation tasks.

Table 3. Experiments on Resnets of varying sizes using BSS as a baseline on the Cars dataset using 15% training data.

Model	**Top-1 Accuracy(%)**		
	BSS	**BSS+TSRS**	**Δ**
Resnet18	38.23±0.14	38.73±0.17	+0.50
Resnet50	44.00±0.32	47.22±0.23	+3.22
Resnet152	**46.31±0.26**	**51.12±0.07**	**+4.81**

Table 4. TSRS embedded into TransUNet using vanilla fine-tuning as the baseline for segmentation. Metric: DSC, HD.

Method	**Synapse multi-organ**		**ISIC2018**		**Spine**	
	DSC↑	**HD↓**	**DSC↑**	**HD↓**	**DSC↑**	**HD↓**
TransUNet$_{base}$	77.16	33.17	88.35	16.07	59.52	5.64
TransUNet$_{base}$+TSRS	**78.30**	**29.28**	**88.86**	**15.18**	**60.23**	**5.24**

Table 5. TSRS embedded into DETR using vanilla fine-tuning as the baseline for object detection. Metric: AP.

Method	**LIDC training data sampling rate**			
	15%	**30%**	**50%**	**100%**
DETR$_{base}$	53.26	60.26	65.88	69.53
DETR$_{base}$+TSRS	**57.02**	**63.91**	**68.16**	**71.33**

Detection. Object detection is another common computer vision task. We used the LIDC dataset [2] and adopted vanilla fine-tuning DETR [5] pre-trained on COCO as the baseline, to investigate the impact of TSRS on fine-tuning for object detection tasks. The other experimental settings were consistent with the main experiment. Tab. 5 presents the results at different training set proportions using the average precision (AP) as the metric. The table revealed that TSRS can enhance fine-tuning performance in object detection tasks, particularly in scenarios with limited training data.

5.6. A case study and discussions

Rifai *et al.* [46] have conducted a study that showed that adding noise only (ANO) to the input can serve as a form of regularization. To better comprehend the performance enhancement brought about by TSRS, we performed experiments to analyze the changes in model robustness and

[2]https://www.synapse.org/#!Synapse:syn3193805/wiki/217789
[3]https://aistudio.baidu.com/datasetdetail/81211

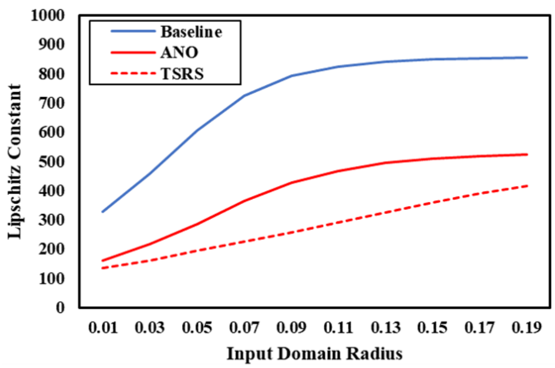
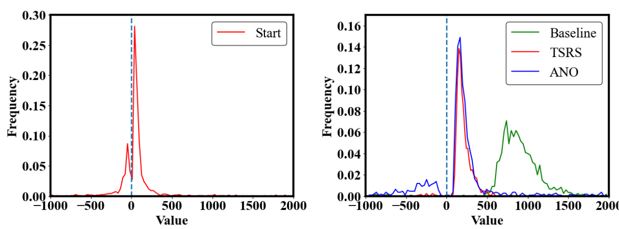


Figure 6. The Lipschitz constant of the MNIST dataset trained on a 3-layer MLP. X-axis: the input domain radius. TSRS has the lowest Lipschitz bound, implying better robustness.



(a) Before Fine-tuning

(b) After Fine-tuning

Figure 7. Hessian max eigenvalue spectra under three conditions: baseline, ANO, and TSRS. Specifically, we calculated the top-5 largest eigenvalues in each mini-batch.

convexity before and after the application of regularization. These changes are usually observed in the first-order and second-order gradients of the model weights.

Robustness. To investigate the model's robustness, we introduce the concept of the Lipschitz constant, which represents the upper bound of the first-order gradient of the model. Generally, a smaller Lipschitz constant indicates better robustness and generalization of the model. To estimate the Lipschitz constant of the model, we adopted the calculation method proposed by *et al.* [49]. We conducted experiments on a 3-layer MLP using the MNIST dataset [30], and Fig. 6 shows the calculated Lipschitz constant of the models obtained by the baseline, ANO, and TSRS under different input domain radius. Both ANO and TSRS reduced the Lipschitz constant of the model, but our method achieved a lower Lipschitz bound, especially when the input domain radius was large. This indicates that our method can obtain a more robust model.

Convexity and Smoothness. We use the eigenvalues of the Hessian matrix as an indicator to analyze the convexity and smoothness of the model. The Hessian matrix represents the second-order gradient matrix of the model. A positive eigenvalue of the Hessian matrix indicates convexity, while a negative eigenvalue indicates non-convexity. The smaller the value of the Hessian matrix, the smoother the loss landscape, and the better the performance and generalization of the model [9, 26, 31]. To conduct our experiments, we fine-tuned the ViT-B model on the CUB dataset using weights pre-trained on ImageNet and computed the Hessian eigenvalues using the method proposed by Park *et al.* [44]. We compared the vanilla fine-tuning, ANO, and TSRS in Fig. 7.

Fig. 7 illustrates the density of the top-5 largest Hessian eigenvalue in each mini-batch. Before fine-tuning, negative Hessian eigenvalues were present, indicating a non-convex loss landscape, consistent with the findings of Park *et al.* [45]. After fine-tuning, negative eigenvalues were suppressed. However, an increase in the eigenvalues of the Hessian is induced, indicating a sharper loss landscape, which can hinder the optimization and generalization of the model. When noise was added to the input, although large Hessian eigenvalues were suppressed, some negative eigenvalues were still retained, suggesting a non-convex loss landscape. This implies that the model may be optimized to a saddle point. In contrast, our fine-tuning method effectively suppressed both large and negative Hessian eigenvalues, indicating improved convexity and smoothness of the loss landscape, leading to better model performance.

6. Conclusion

In this paper, we conduct an analysis of the fine-tuning task through the lens of structural risk minimization (SRM). We recognize the inherent need for a neural network to have a high complexity to accommodate the large-scale pre-training source data while requiring a lower complexity to align with the limited amount of target data in order to minimize structural risk. However, conventional fine-tuning methods fail to effectively reduce the complexity of the pre-trained model, resulting in an increase in structural risk and a negative impact on generalization ability. Based on these insights, we propose a simple yet effective regularization approach, Tuning Stable Rank Shrinkage (TSRS), to effectively constrain the model complexity during fine-tuning. By addressing the previously overlooked aspect of SRM in fine-tuning, TSRS can be seamlessly embedded into existing fine-tuning techniques, leading to additional performance improvements. This work serves as an inspiration for researchers aiming to leverage pre-trained large models for target tasks with limited data. However, in scenarios where the volume of the source and target domains are similar, our method may not yield significant improvements since there is no need to constrain the structural risk.

Acknowledgement This work is supported by the National Natural Science Foundation in China under Grant 62371016 and U23B2063, the Bejing Natural Science Foundation Haidian District Joint Fund in China under Grant L222032.

References

[1] Ethem Alpaydin. *Machine learning*. Mit Press, 2021. 2

[2] Samuel G Armato III, Geoffrey McLennan, Luc Bidaut, Michael F McNitt-Gray, Charles R Meyer, Anthony P Reeves, Binsheng Zhao, Denise R Aberle, Claudia I Henschke, Eric A Hoffman, et al. The lung image database consortium (lidc) and image database resource initiative (idri): a completed reference database of lung nodules on ct scans. *Medical physics*, 38(2):915–931, 2011. 7

[3] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning*, pages 254–263. PMLR, 2018. 3, 4, 1

[4] José-Ramón Cano. Analysis of data complexity measures for classification. *Expert systems with applications*, 40(12): 4820–4831, 2013. 1

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 7

[6] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021. 7

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2

[8] Xinyang Chen, Sinan Wang, Bo Fu, Mingsheng Long, and Jianmin Wang. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 2, 3, 5, 6

[9] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pre-training or strong data augmentations. *arXiv preprint arXiv:2106.01548*, 2021. 8

[10] Noel Codella, Veronica Rotemberg, Philipp Tschandl, M Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, et al. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368*, 2019. 7

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[12] Michele Donini, Luca Oneto, Shai Ben-David, John S Shawe-Taylor, and Massimiliano Pontil. Empirical risk minimization under fairness constraints. *Advances in neural information processing systems*, 31, 2018. 2

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[14] RCNN Faster. Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 9199(10.5555):2969239–2969250, 2015. 2

[15] Patrick Glandorf, Timo Kaiser, and Bodo Rosenhahn. Hypersparse neural networks: Shifting exploration to exploitation through adaptive regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1234–1243, 2023. 6

[16] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015. 2

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[18] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4918–4927, 2019. 1, 2

[19] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 1, 2

[20] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2

[21] Xia Hu, Weiqing Liu, Jiang Bian, and Jian Pei. Measuring model complexity of neural networks with curve activation functions. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1521–1531, 2020. 1

[22] Xia Hu, Lingyang Chu, Jian Pei, Weiqing Liu, and Jiang Bian. Model complexity of deep learning: A survey. *Knowledge and Information Systems*, 63:2585–2619, 2021. 3

[23] Hang Hua, Xingjian Li, Dejing Dou, Chengzhong Xu, and Jiebo Luo. Noise stability regularization for improving bert fine-tuning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3229–3241, 2021. 3

[24] Junguang Jiang, Yang Shu, Jianmin Wang, and Mingsheng Long. Transferability in deep learning: A survey, 2022. 5

[25] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019. 3

[26] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016. 8

[27] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan,

John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526, 2017. 1, 2, 3

[28] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013. 5, 6

[29] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 7

[30] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. 8

[31] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018. 8

[32] Ling Li. *Data complexity in machine learning and novel classification algorithms*. California Institute of Technology, 2006. 1

[33] Xingjian Li, Haoyi Xiong, Hanchao Wang, Yuxuan Rao, Liping Liu, Zeyu Chen, and Jun Huan. Delta: Deep learning transfer using feature map with attention for convolutional networks. *arXiv preprint arXiv:1901.09229*, 2019. 1, 2, 6

[34] Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In *The 22nd international conference on artificial intelligence and statistics*, pages 888–896. PMLR, 2019. 3

[35] Tailin Liang, John Glossner, Lei Wang, Shaobo Shi, and Xiaotong Zhang. Pruning and quantization for deep neural network acceleration: A survey. *Neurocomputing*, 461:370–403, 2021. 2

[36] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744, 2017. 2

[37] Ziquan Liu, Yi Xu, Yuanhong Xu, Qi Qian, Hao Li, Xiangyang Ji, Antoni Chan, and Rong Jin. Improved fine-tuning by better leveraging pre-training data. *Advances in Neural Information Processing Systems*, 35:32568–32581, 2022. 1, 2, 6

[38] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 5, 6

[39] Song Mei, Yu Bai, and Andrea Montanari. The landscape of empirical risk for nonconvex losses. *The Annals of Statistics*, 46(6A):2747–2774, 2018. 2

[40] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017. 3

[41] Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017. 4

[42] Andrew Y Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78, 2004. 6

[43] Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018. 3, 1

[44] Namuk Park and Songkuk Kim. Blurs behave like ensembles: Spatial smoothings to improve accuracy, uncertainty, and robustness. In *International Conference on Machine Learning*, pages 17390–17419. PMLR, 2022. 8

[45] Namuk Park and Songkuk Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022. 8

[46] Salah Rifai, Xavier Glorot, Yoshua Bengio, and Pascal Vincent. Adding noise to the input of a model trained with a regularized objective. *arXiv preprint arXiv:1104.3250*, 2011. 7

[47] Mark Rudelson and Roman Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)*, 54(4):21–es, 2007. 3

[48] Amartya Sanyal, Philip HS Torr, and Puneet K Dokania. Stable rank normalization for improved generalization in neural networks and gans. *arXiv preprint arXiv:1906.04659*, 2019. 3, 6

[49] Zhouxing Shi, Yihan Wang, Huan Zhang, J Zico Kolter, and Cho-Jui Hsieh. Efficiently computing local lipschitz constants of neural networks via bound propagation. *Advances in Neural Information Processing Systems*, 35:2350–2364, 2022. 8

[50] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global, 2010. 1, 2, 3

[51] Vladimir Vapnik. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4, 1991. 1, 2, 3

[52] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999. 2

[53] LN Vasershtein. Stable rank of rings and dimensionality of topological spaces. *Functional Analysis and its Applications*, 5(2):102–110, 1971. 3

[54] Diego Vidaurre, Concha Bielza, and Pedro Larranaga. A survey of l1 regression. *International Statistical Review*, 81 (3):361–387, 2013. 6

[55] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 4, 5, 6

[56] Yi Wang, Zhen-Peng Bian, Junhui Hou, and Lap-Pui Chau. Convolutional neural networks with dynamic regularization. *IEEE Transactions on Neural Networks and Learning Systems*, 32(5):2299–2304, 2020. 6

[57] LI Xuhong, Yves Grandvalet, and Franck Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *International Conference on Machine Learning*, pages 2825–2834. PMLR, 2018. 1, 2

[58] Ke Yan, Xiaosong Wang, Le Lu, and Ronald M Summers. Deeplesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning. *Journal of medical imaging*, 5(3):036501–036501, 2018. 1

[59] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014. 5, 2

[60] Kaichao You, Zhi Kou, Mingsheng Long, and Jianmin Wang. Co-tuning for transfer learning. *Advances in Neural Information Processing Systems*, 33:17236–17246, 2020. 1, 2, 5, 6

[61] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 1

[62] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 2

[63] Wen Zhang, Lingfei Deng, Lei Zhang, and Dongrui Wu. A survey on negative transfer. *IEEE/CAA Journal of Automatica Sinica*, 2022. 1

[64] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020. 1, 2