RL-Tune: A Deep Reinforcement Learning Assisted Layer-wise Fine-Tuning Approach for Transfer Learning

Tanvir Mahmud [1] **Natalia A. Frumkin** [1] **Diana Marculescu** [1]

Abstract

Data scarcity is one of the major challenges in many real-world applications. To handle low-data regimes, practitioners often take an existing pre-trained network and fine-tune it on a data-deficient target task. In this setup, a network is pre-trained on a source dataset and fine-tuned on a different, potentially smaller, target dataset. We address two critical challenges with transfer learning via fine-tuning: (1) The required amount of fine-tuning greatly depends on the distribution shift from source to target dataset. (2) This distribution shift greatly varies by layer, thereby requiring layer-wise adjustments in fine-tuning to adapt to this distribution shift while preserving the pre-trained network's feature representation. To overcome these challenges, we propose *RL-Tune*, a layer-wise fine-tuning framework for transfer learning which leverages reinforcement learning to adjust learning rates as a function of the target data shift. In our RL framework, the *state* is a collection of the intermediate feature activations generated from training samples. The agent generates layer-wise learning rates as *actions* for fine-tuning based on the current state and obtains sample accuracy as the *reward*. *RL-Tune* outperforms other state-of-the-art approaches on standard transfer learning benchmarks by a large margin, *e.g.*, 6% mean accuracy improvement on CUB-200-2011 with 15% data.

1. Introduction

In low-data regimes, neural networks suffer from overfitting and poor generalization on unseen training samples (Bansal et al., 2020). One effective solution to learning with insufficient data is *transfer learning via fine-tuning* where a model is pre-trained on the source dataset and subsequently fine-tuned on the target task (Tan et al., 2018). We have seen the effectiveness of transfer learning via fine-tuning on a wide variety of applications including medical imaging (Tajbakhsh et al., 2016) and standard ML benchmarks (Guo et al., 2019). Prior work has shown that layer-wise fine-tuning is beneficial for transfer learning, because the pre-trained model's initial layers act as general feature extractors whereas the final layers can extract more specific features (Yosinski et al., 2014; Girshick et al., 2014). When the source and target domains share similar features, we can leverage the model's general feature extractor and only fine-tune the layers where the target task's features differ significantly.

In traditional fine-tuning approaches, we either fine-tune globally, or freeze some layers and fine-tune the rest with a uniform learning rate. Once a model is pre-trained, its earlier layers do not need as much fine-tuning since they act as general feature extractors (Sun et al., 2019; Ro & Choi, 2021). For transfer learning applications, there is a variety of techniques to apply layer-specific fine-tuning, including heuristic-based methods (Kornblith et al., 2019; Li et al., 2020) and supervised approaches, such as probabilistic frameworks (You et al., 2020) and a zoo of models (Shu et al., 2021). In contrast to the pre-training phase, the layer-wise learning rates in the transfer phase are more sensitive to the data distribution shift between the source and target datasets. Improper learning rate selection may lead to catastrophic forgetting that neutralizes the advantages of transfer learning (Sun et al., 2019).

Several approaches have been explored to automate the hyper-parameter search in the training phase. To tune the hyper-parameters, a population based bandit algorithm is found to be effective in optimizing RL training (Parker-Holder et al., 2020). A meta learning approach has been introduced recently (Oh et al., 2020) for learning the update rule of the RL algorithm. However, the data distribution shift is a core problem for most transfer learning applications (Weiss et al., 2016). With significant distributional shift, transfer learning becomes difficult as the source and target features may not match. *To overcome this challenge,*

[1]Department of Electrical and Computer Engineering, The University of Texas at Austin, Texas, USA. Correspondence to: Tanvir Mahmud <tanvirmahmud@utexas.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

we propose RL-Tune which leverages reinforcement learning to improve knowledge transfer through sample-based layer-wise fine-tuning (see Fig. 1(a)).

In our RL setup, the agent's input *state* is the collection of all intermediary feature maps, which captures the domain shift on a per-sample basis. In response to the current state, the agent's *action* is a vector of learning rates for layer-wise fine-tuning. Once one round of fine-tuning is complete, the agent's *reward* is the corresponding sample accuracy. We carried out extensive experiments and find that *RL-Tune* outperforms state-of-the-art approaches (You et al., 2020; Li & Hoiem, 2018; Chen et al., 2019; Li et al., 2018) by more than 2% for a variety of sampling rates on the CUB-200-2011, Stanford, and FGVC Aircraft datasets.

2. Related Work

2.1. Pre-training

Training very deep models on large-scale benchmark datasets (Deng et al., 2009; Lin et al., 2014) establishes several well known architectures, such as AlexNet (Krizhevsky et al., 2012), ResNet (He et al., 2016), DenseNet (Huang et al., 2017), Vision Transformers (Han et al., 2022), and BERT (Devlin et al., 2018). The pre-trained weights of these deep networks are widely used for transfer learning on similar datasets with significantly lower amount of data. Most of these networks are trained with supervised learning on large-scale labeled data. To leverage large-scale unlabeled data, semi-supervised (Chen et al., 2020) and self-supervised (Hendrycks et al., 2019) learning have been adopted to increase the diversity of pre-trained models (Radford et al., 2021; Brown et al., 2020).

2.2. Transfer learning

Fine-tuning deep pre-trained model was established as an effective approach for better performance (Girshick et al., 2014; Agrawal et al., 2014) and faster convergence (He et al., 2019) particularly in low-data regimes. (Li et al., 2020) confirms that the choice of hyper-parameters, particularly learning rates and momentum, play a significant role in the final performance of transfer learning. Various approaches have been explored for better regularization of the transfer learning with effective hyper-parameter selection. (Kornblith et al., 2019) proposes a grid-search based approach to search for better hyper-parameters. (Li et al., 2020) provides an elaborate guideline of learning rates and other hyper-parameter selections that confirms the increasing challenges of fine-tuning with dissimilar target datasets. However, all of these approaches primarily focus on manual search of hyper-parameters that hardly take care of the distribution shift of the target datasets and various abstractions of intermediate levels. Apart from these, Co-Tuning (You et al., 2020) attempts to integrate the whole pre-trained network, including top fully connected layers, by learning the probabilistic class mapping of source and target data. Self-tuning (Wang et al., 2021) integrates the additional unlabeled data with semi-supervised learning into transfer learning. Zoo-Tuning (Shu et al., 2021) learns the combinations of a zoo of pre-trained models for improved transfer learning. However, most of these approaches operate with a pre-defined set of hyper-parameters. We particularly focus on improving standard transfer learning from a pre-trained model to a target task with limited amount of labeled data.

3. Methodology

3.1. Problem Formulation

Given a pre-trained model $\mathcal{M}_0$ on a source dataset $D_s = \{(x_s^i, y_s^i\}_{i=1}^{m_s}$ where x_s, y_s, m_s represent the source image, labels, and source dataset size, respectively, we generate the fine-tuned model $\mathcal{M}_f$ on the target dataset $D_g = \{(x_g^i, y_g^i\}_{i=1}^{m_g}$ where x_g, y_g, m_g represent the target image, labels, and target dataset size, respectively. Only the pre-trained model $\mathcal{M}_0$, and the target dataset D_g are available for transfer learning.

3.2. Proposed Reinforcement Learning Framework

We begin with the target model initialized with the pre-trained weights and deploy an RL agent to adapt layer-wise learning rates as fine-tuning progresses (Fig. 1(a)). The RL framework is defined as a tabular RL problem by specifying the state, action, reward, and next state as $(s_t, a_t, r_{t+1}, s_{t+1})$. We primarily consider policy gradient algorithms that require no knowledge of the underlying environment dynamics function (Schulman et al., 2017).

State space

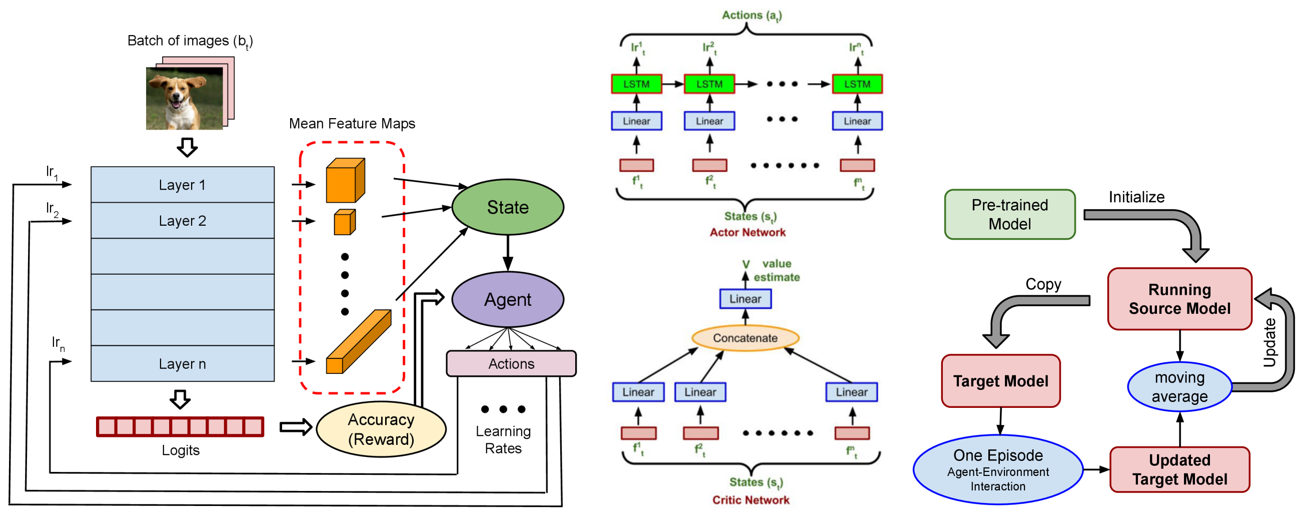
The state space (S) corresponds the current representation of the model for target data distribution. A batch (b_t) is fed into the target model and the mean of intermediate feature activations f^i is extracted from each layer i. The state representation at each timestamp t ($s_t \in S$) incorporates all the mean feature activations (f^i) of intermediate layers; that can be represented by:

$$s_t = \{f_t^1, f_t^2, \dots, f_t^{n-1}, f_t^n\} \quad (1)$$

where f_t^i represents the feature map obtained from i^{th} layer at timestamp t from all n layers of the target model.

Action Space

The action space (A) represents the set of learning rates for each layer in the sequential fine-tuning process. The agent considers the current feature representations of each



(a) *RL-Tune* Framework (b) Agent Architecture (c) Episodic Moving Average Control

Figure 1. The proposed approach for transfer learning via layer-wise fine-tuning. (a) *RL-Tune* framework: The RL agent receives each layer's mean feature maps (averaged over the batch), and outputs one learning rate per layer. (b) Agent architecture: An actor-critic model built on linear and LSTM layers. (c) Episodic Moving Average Control: The target model is trained with agent-environment interaction while the running source model is updated with exponential moving averages of the updated target model.

layer (s_t) obtained with the sample batch (b_t) and predicts the learning rate of each layer (a_t) to update the model for generating new state representation (s_{t+1}). The action (a_t) generated at each time stamp t can be represented by

$$a_t(s_t) = \{lr^1_t, lr^2_t, \dots, lr^{n-1}_t, lr^n_t\} \tag{2}$$

where lr^i_t represents the learning rate of the i^{th} layer at time step t. The learning rate of each layer can be chosen either from a discrete or continuous set of choices.

Reward

The reward function (r_{t+1}) represents the immediate gain achieved through performing an action (a_t) based on the current (s_{t+1}) and past state (s_t). We update the target model based on the learning rates (a_t) with stochastic gradient descent on current batch of samples (b_t). The reward at time t (r_{t+1}) is represented by the accuracy of the updated target model ($\mathcal{M}$') on the current sample batch (b_t), which is given by

$$r_{t+1}(s_t, a_t, s_{t+1}) = Accuracy(\mathcal{M}', b_t) \tag{3}$$

Objective Definition

We define our RL problem as an episodic task in the sequential fine-tuning scheme. The primary objective is to derive a better policy for generating layer-wise learning rates over the old policy on the current model/state representations from observed interactions. We primarily focus on the proximal policy objective (Schulman et al., 2017) L_t at each time-step t which is defined as the expectation ($\mathbb{E}_t$) of the policy improvement as:

$$L_t(\theta) = \hat{\mathbb{E}}_t \left[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right] \tag{4}$$

where θ represents the policy parameters, $\hat{A}_t$ represents the estimated advantage function corresponding to the reward, ϵ is a hyper-parameter (usually 0.1) and $r_t(\theta)$ is the ratio of the action probability under the new and old policies.

3.3. Proposed Actor-Critic Agent Architecture

The agent architecture consists of an actor-critic model in the proposed formulation (Fig. 1(b)). The actor network generates the action choices (a_t) representing layer-wise learning rates based on the corresponding state (s_t) and current policy. It consists of an intermediate linear layer to process the mean feature activations ($f^1_t, f^2_t, \dots, f^{n-1}_t, f^n_t$) representing state followed by a long short term memory (LSTM) layer to predict the sequence of learning rates for each layer. The MLP-based critic network estimates the value (v_t) of the current state (s_t).

3.4. Proposed Episodic Moving Average Control

Inspired by the moving average baseline in (Zoph & Le, 2016), the model is updated through a moving average of the weights in sequential episodes (Fig. 1(c)). Rather than directly updating the target model in one episode, we gradually update the weights with a moving average which encourages more agent-environment interactions. Firstly,

Table 1. Performance comparison of state-of-the-art approaches with pre-trained ResNet-50 on ImageNet (Deng et al., 2009)

Dataset	Method	Sampling rates			
		15%	30%	50%	100%
CUB-200-2011	Fine-Tune (baseline)	45.25 ± 0.12	59.28 ± 0.21	70.12 ± 0.16	78.01 ± 0.16
	L^2-SP (Li & Hoiem, 2018)	45.08 ± 0.19	57.78 ± 0.24	69.47 ± 0.29	78.44 ± 0.17
	DELTA (Li et al., 2018)	46.83 ± 0.21	60.37 ± 0.25	71.38 ± 0.20	78.63 ± 0.18
	BSS (Chen et al., 2019)	47.74 ± 0.23	63.38 ± 0.29	72.56 ± 0.17	78.85 ± 0.31
	Co-Tuning (You et al., 2020)	52.58 ± 0.53	66.47 ± 0.17	74.64 ± 0.36	81.24 ± 0.14
	***RL-Tune* (Ours)**	$\mathbf{58.75 \pm 0.18}$	$\mathbf{71.16 \pm 0.21}$	$\mathbf{78.92 \pm 0.25}$	$\mathbf{84.12 \pm 0.22}$
Stanford Cars	Fine-Tune (baseline)	36.77 ± 0.12	60.63 ± 0.18	75.10 ± 0.21	87.20 ± 0.19
	L^2-SP (Li & Hoiem, 2018)	36.10 ± 0.30	60.30 ± 0.28	75.48 ± 0.22	86.58 ± 0.26
	DELTA (Li et al., 2018)	39.37 ± 0.34	63.28 ± 0.27	76.53 ± 0.24	86.32 ± 0.20
	BSS (Chen et al., 2019)	40.57 ± 0.12	64.13 ± 0.18	76.78 ± 0.21	87.63 ± 0.27
	Co-Tuning (You et al., 2020)	46.02 ± 0.18	69.09 ± 0.10	80.66 ± 0.25	89.53 ± 0.09
	***RL-Tune* (Ours)**	$\mathbf{52.19 \pm 0.24}$	$\mathbf{75.28 \pm 0.23}$	$\mathbf{84.24 \pm 0.26}$	$\mathbf{91.81 \pm 0.22}$
FGVC Aircraft	Fine-Tune (baseline)	39.57 ± 0.20	57.46 ± 0.12	67.93 ± 0.28	81.13 ± 0.21
	L^2-SP (Li & Hoiem, 2018)	39.27 ± 0.24	57.12 ± 0.27	67.46 ± 0.26	80.98 ± 0.29
	DELTA (Li et al., 2018)	42.16 ± 0.21	58.60 ± 0.29	68.51 ± 0.25	80.44 ± 0.20
	BSS (Chen et al., 2019)	40.41 ± 0.12	59.23 ± 0.31	69.19 ± 0.13	81.48 ± 0.18
	Co-Tuning (You et al., 2020)	44.09 ± 0.67	61.65 ± 0.32	72.73 ± 0.08	83.87 ± 0.09
	***RL-Tune* (Ours)**	$\mathbf{50.11 \pm 0.23}$	$\mathbf{66.63 \pm 0.19}$	$\mathbf{76.77 \pm 0.25}$	$\mathbf{86.12 \pm 0.15}$

the pre-trained model is initialized as the source model, and the target model starts with the source model weights before each episode. One episode continues for several epochs with the replay buffer tracking the agent-environment interactions, and the target model is updated with stochastic gradient descent. The agent model is updated after each epoch in an episode based on the stored interactions in the replay buffer. After each episode, the weights of the source model are updated based on the moving average of the prior weights and the weights of the updated target model.

4. Results and Discussion

4.1. Datasets

We consider three publicly available benchmark datasets for image classifications: CUB-200-2011 (Welinder et al., 2010) (11,788 images for 200 bird classes), Stanford-cars (Krause et al., 2013) (16,185 images with 196 classes), and FGVC Aircraft (Maji et al., 2013) (10,000 images with 100 aircraft classes). We have experimented with different sampling rates to analyze the effect on low-data regime following prior work (You et al., 2020). Unlike Self-Tuning (Wang et al., 2021) that leverages the remaining data as unlabeled samples for semi-supervised learning, only the sampled data is used following standard transfer learning.

4.2. Implementation Details

All the experiments have been conducted with OpenAI gym-toolkit and official PyTorch framework. Though the proposed *RL-Tune* can be modeled with either continuous or discrete actions, we consider the discrete action setup with 20 intermediate uniform steps for the learning rate of each layer (minimum 0, maximum 0.01). We have used the Adam optimizer for all experiments where the learning rate of each layer is adapted with the RL-Tune framework. We have considered ten epochs per-episodes in all experiments of *RL-Tune*. Moreover, the experiments continued for five episodes with an exponential moving average factor of 0.2. We considered the ResNet-50 model for experiments pre-trained on large-scale ImageNet dataset (Deng et al., 2009). For generating the baseline, we have followed the standard process of fine-tuning with uniform learning rate over the pretrained network with Adam optimizer. Each experiment has been carried out at least three times to get the expected value of the reported accuracy.

4.3. Main Results

We have compared our method with several state-of-the-art approaches on three benchmark datasets as summarized in Table 1. *RL-Tune* consistently outperforms its state-of-the art counterpart *Co-Tuning* (You et al., 2020) with average improvements of 2.72% on CUB-200-2011, 2.28% on Stanford Cars, and 2.25% on FGVC Aircraft with 100% training data. Moreover, in all other state-of-the-art approaches, uniform learning rate is used over the network in fine-tuning where several approaches to adapt the distribution shift of the source and target dataset is incorporated, *e.g. Co-Tuning* introduces the learning of the category relationship between

Table 2. The effect of the number of episodes on training accuracy with 50% sampling rate. The performance gradually improves with more episodes in the exponential moving average approach.

No. of Episodes	CUB-200-2011	Stanford Cars	FGVC Aircraft
1	76.12 ± 0.21	81.91 ± 0.22	67.93 ± 0.28
2	77.08 ± 0.17	82.77 ± 0.16	67.46 ± 0.26
3	78.35 ± 0.24	83.52 ± 0.18	68.51 ± 0.25
4	78.78 ± 0.11	84.11 ± 0.14	72.73 ± 0.08
5	$\mathbf{78.92 \pm 0.25}$	$\mathbf{84.24 \pm 0.26}$	$\mathbf{76.77 \pm 0.25}$

the target dataset and the source dataset to adapt accordingly. However, the proposed approach particularly controls learning of each layer to adapt the fine-tuning process on the target dataset. In low-data regimes using only a fraction of the target dataset, we see even sharper improvements over baseline methods. For example, *RL-Tune* achieves 6.02% higher accuracy on FGVC Aircraft with only 15% of training samples, and 4.98% improvements with 30% training data. This demonstrates that *RL-Tune* can be very effective for small datasets when only a few thousand samples are available, and strictly improves over Co-Tuning in all sampling regimes.

5. Ablation Studies

5.1. Ablation of the Number of Episodes

In the proposed exponential moving average approach (see Section 3.4), the total number of training episodes plays an important role on the final performance. We have carried out an ablation study on the choice of total number of episodes with a moving average factor of 0.2. The final accuracy obtained on different datasets with 50% sampling rate is summarized in Table 2. Each episode is continued for ten epochs. The performance improves by incorporating more episodes, representing more interaction with the training environment. The best performance is achieved for a total five episodes. However, the improvement margin gradually shrinks with increasing number of episodes.

5.2. Ablation of the Number of Epochs Per-Episode

Each episode is continued for a number of epochs that represents the agent-environment interactions carried out before the exponential moving average updates (see Section 3.4). The performance obtained with different number of training epochs per episode is provided in Table 3. Each run is continued for five episodes with an exponential moving average factor of 0.2. Similar to the total number of episodes, the performance continues to improve with increasing number of training epochs. However, the agent adapts quickly with the optimization dynamics and the improvement gets saturated gradually.

Table 3. The effect of the number of training epoch per episode with 5 episodes run and 50% sampling rate. Additional epochs during each episode improve performance.

No. of Epochs	CUB-200-2011	Stanford Cars	FGVC Aircraft
1	73.56 ± 0.13	78.93 ± 0.32	71.11 ± 0.21
5	76.87 ± 0.21	83.22 ± 0.19	74.31 ± 0.17
8	78.10 ± 0.23	84.02 ± 0.18	75.93 ± 0.23
10	$\mathbf{78.92 \pm 0.25}$	$\mathbf{84.24 \pm 0.26}$	$\mathbf{76.77 \pm 0.25}$

Table 4. The effect of the total number of learning rate steps for each layer with 50% sampling rate. Best performance is achieved with an intermediate value of 20 steps.

Learning Rate Steps	CUB-200-2011	Stanford Cars	FGVC Aircraft
10	77.78 ± 0.18	82.38 ± 0.20	75.94 ± 0.27
15	78.43 ± 0.24	83.98 ± 0.28	76.36 ± 0.21
20	$\mathbf{78.92 \pm 0.25}$	84.24 ± 0.26	$\mathbf{76.77 \pm 0.25}$
25	78.17 ± 0.29	$\mathbf{84.43 \pm 0.32}$	76.56 ± 0.26

5.3. Ablation of the Per-layer Learning-rate Steps

Since we considered a discrete set of actions representing learning rate for each layer (see Section 3.2), the choice of total number of intermediate steps for learning rates significantly impacts the final performance. We have carried out an ablation study on different choices of layer-wise learning rates with a maximum of 0.01 and minimum of 0. The choices are distributed uniformly in the specific range of learning rates. The performance obtained with different learning rates is presented in Table 4. We note that a lower number of steps achieves lower accuracy whereas a larger number of steps increases complexity. The best performance is achieved with an intermediate choice of 20 steps.

6. Conclusion

We propose *RL-Tune*, a novel reinforcement learning based framework for determining layer-wise learning rates in transfer learning. *RL-Tune* adapts to the domain shift by adjusting learning rates for each layer as the network state changes. We apply fine-grained adjustments via layer-wise learning rates, which aim to preserve the pre-trained network's deductive power while allowing each layer to adapt to the target distribution. We have shown significant improvement in low-data regimes for three benchmark datasets, illustrating the effectiveness of RL-based learning rate selection for transfer learning.

Acknowledgements

This research was supported in part by the Office of Naval Research, Minerva Program, and a UT Cockrell School of Engineering Doctoral Fellowship.

References

Agrawal, P., Girshick, R., and Malik, J. Analyzing the performance of multilayer neural networks for object recognition. In European conference on computer vision, pp. 329–344. Springer, 2014.

Bansal, M. A., Sharma, D. R., and Kathuria, D. M. A systematic review on data scarcity problem in deep learning: Solution and applications. ACM Computing Surveys (CSUR), 2020.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. Advances in neural information processing systems, 33: 1877–1901, 2020.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. Big self-supervised models are strong semi-supervised learners. Advances in neural information processing systems, 33:22243–22255, 2020.

Chen, X., Wang, S., Fu, B., Long, M., and Wang, J. Catastrophic forgetting meets negative transfer: Batch spectral shrinkage for safe transfer learning. Advances in Neural Information Processing Systems, 32, 2019.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pp. 248–255. Ieee, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587, 2014.

Guo, Y., Shi, H., Kumar, A., Grauman, K., Rosing, T., and Feris, R. Spottune: transfer learning through adaptive fine-tuning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 4805–4814, 2019.

Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., et al. A survey on vision transformer. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778, 2016.

He, K., Girshick, R., and Dollár, P. Rethinking imagenet pre-training. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4918–4927, 2019.

Hendrycks, D., Mazeika, M., Kadavath, S., and Song, D. Using self-supervised learning can improve model robustness and uncertainty. Advances in Neural Information Processing Systems, 32, 2019.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708, 2017.

Kornblith, S., Shlens, J., and Le, Q. V. Do better imagenet models transfer better? In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2661–2671, 2019.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In Proceedings of the IEEE international conference on computer vision workshops, pp. 554–561, 2013.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 2012.

Li, H., Chaudhari, P., Yang, H., Lam, M., Ravichandran, A., Bhotika, R., and Soatto, S. Rethinking the hyperparameters for fine-tuning. arXiv preprint arXiv:2002.11770, 2020.

Li, X., Xiong, H., Wang, H., Rao, Y., Liu, L., and Huan, J. Delta: Deep learning transfer using feature map with attention for convolutional networks. In International Conference on Learning Representations, 2018.

Li, Z. and Hoiem, D. Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence, 40(12):2935–2947, 2018.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In European conference on computer vision, pp. 740–755. Springer, 2014.

Maji, S., Rahtu, E., Kannala, J., Blaschko, M., and Vedaldi, A. Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151, 2013.

Oh, J., Hessel, M., Czarnecki, W. M., Xu, Z., van Hasselt, H. P., Singh, S., and Silver, D. Discovering reinforcement learning algorithms. Advances in Neural Information Processing Systems, 33:1060–1070, 2020.

Parker-Holder, J., Nguyen, V., and Roberts, S. J. Provably efficient online hyperparameter optimization with population-based bandits. Advances in Neural Information Processing Systems, 33:17200–17211, 2020.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In International Conference on Machine Learning, pp. 8748–8763. PMLR, 2021.

Ro, Y. and Choi, J. Y. Autolr: Layer-wise pruning and auto-tuning of learning rates in fine-tuning of deep networks. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pp. 2486–2494, 2021.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

Shu, Y., Kou, Z., Cao, Z., Wang, J., and Long, M. Zoo-tuning: Adaptive transfer from a zoo of models. In International Conference on Machine Learning, pp. 9626–9637. PMLR, 2021.

Sun, C., Qiu, X., Xu, Y., and Huang, X. How to fine-tune bert for text classification? In China national conference on Chinese computational linguistics, pp. 194–206. Springer, 2019.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., and Liang, J. Convolutional neural networks for medical image analysis: Full training or fine tuning? IEEE transactions on medical imaging, 35(5):1299–1312, 2016.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. A survey on deep transfer learning. In International conference on artificial neural networks, pp. 270–279. Springer, 2018.

Wang, X., Gao, J., Long, M., and Wang, J. Self-tuning for data-efficient deep learning. In International Conference on Machine Learning, pp. 10738–10748. PMLR, 2021.

Weiss, K., Khoshgoftaar, T. M., and Wang, D. A survey of transfer learning. Journal of Big data, 3(1):1–40, 2016.

Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., and Perona, P. Caltech-ucsd birds 200. 2010.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? Advances in neural information processing systems, 27, 2014.

You, K., Kou, Z., Long, M., and Wang, J. Co-tuning for transfer learning. Advances in Neural Information Processing Systems, 33:17236–17246, 2020.

Zoph, B. and Le, Q. V. Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578, 2016.