One Step Learning, One Step Review

Xiaolong Huang, Qiankun Li[1, 2*], Xueran Li, Xuesong Gao[3]

[1]Institute of Intelligent Machines, Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei, China
[2]Department of Automation, University of Science and Technology of China, Hefei, China
[3]Chongqing University of Science and Technology, Chongqing, China
{hirox827, xueran.lxr}@gmail.com, qklee@mail.ustc.edu.cn, xuesonggxs@foxmail.com

Abstract

Visual fine-tuning has garnered significant attention with the rise of pre-trained vision models. The current prevailing method, full fine-tuning, suffers from the issue of knowledge forgetting as it focuses solely on fitting the downstream training set. In this paper, we propose a novel weight rollback-based fine-tuning method called OLOR (One step Learning, One step Review). OLOR combines fine-tuning with optimizers, incorporating a weight rollback term into the weight update term at each step. This ensures consistency in the weight range of upstream and downstream models, effectively mitigating knowledge forgetting and enhancing fine-tuning performance. In addition, a layer-wise penalty is presented to employ penalty decay and the diversified decay rate to adjust the weight rollback levels of layers for adapting varying downstream tasks. Through extensive experiments on various tasks such as image classification, object detection, semantic segmentation, and instance segmentation, we demonstrate the general applicability and state-of-the-art performance of our proposed OLOR. Code is available at https://github.com/rainbow-xiao/OLOR-AAAI-2024.

Introduction

With the rapid advancement of deep learning technology, numerous large-scale image datasets have been established (Schuhmann et al. 2022; Russakovsky et al. 2015; Schuhmann et al. 2021), resulting in many promising pre-trained visual models (Radford et al. 2021; He et al. 2022; Bao et al. 2021). These pre-trained models can effectively solve related but distinct visual tasks through transfer learning and fine-tuning techniques (Wu, Sun, and Ouyang 2023; Shen et al. 2021). The fundamental fine-tuning methods are linear probing and full fine-tuning (Zhang, Isola, and Efros 2017). In linear probing, the pre-trained model's backbone is frozen, and only the head specific to the downstream task is trained. However, this approach often restricts the performance of the pre-trained backbone. On the other hand, full fine-tuning involves training the entire network directly, but it usually leads to knowledge forgetting (De Lange et al. 2021).

Rehearsal methods (Rebuffi et al.; Rolnick et al. 2019; Liu et al. 2020; Merlin et al. 2022), based on the replay mechanism, involve retraining on a subset of stored upstream samples while learning new tasks. However, this approach is quite inefficient. EWC (Kirkpatrick et al. 2017) proposes a regularization-based fine-tuning method that uses the Fisher information matrix to determine the importance of weight parameters. This helps adjust the parameters between upstream and downstream tasks, reducing forgetting. L2-SP (Xuhong, Grandvalet, and Davoine 2018) uses an L2 penalty to restrict the updates of parameters, addressing knowledge forgetting during fine-tuning. However, it is not compatible with adaptive optimizers (Loshchilov and Hutter 2017; Guan 2023), which may produce the wrong regularization direction. Parameter isolation methods (Jia et al. 2022; Sohn et al. 2023) create new branches or modules for different network models and tasks for downstream tasks. However, it introduces additional new training parameters, requires certain training skills, and has lower generality than rehearsal methods.

In this paper, we propose a novel fine-tuning method combined with optimizers to solve knowledge forgetting, called OLOR (One step Learning, One step Review). Specifically, OLOR introduces a weight rollback term to the weight update term during the fine-tuning stage, allowing the model to gradually approach the pre-trained weights while learning the downstream task. This process avoids delay defects and makes the weights of the upstream and downstream models more similar. In addition, a layer-wise penalty is devised to employ penalty decay and the diversified decay rate to adjust the weight rollback levels of layers. Penalty decay combines feature pyramids with transfer learning, giving more significant weight rollback to shallow layers related to shallow features such as color and texture, and smaller weight backtracking to deep layers related to deep features such as semantic information. The diversified decay rate is adjusted to enhance applicability according to the variations between up and downstream tasks. OLOR with layer-wise penalty enables each layer of the model to update according to its needs, resulting in superior extraction of generalized features. Finally, OLOR is incorporated into optimizers, thereby introducing negligible extra computational overhead. It also works well with popular optimizers such as Adam (Loshchilov and Hutter 2017; Guan 2023) and SGD (Keskar and Socher 2017), meeting specific needs under various conditions.

*Corresponding authors

Our OLOR fine-tuning method achieves state-of-the-art performance on ten popular visual task datasets covering general classification, fine-grained classification, long-tail classification, cross-domain classification, object detection, semantic segmentation, and instance segmentation. Validation experiments and ablation analysis demonstrate the performance of OLOR in solving the problem of knowledge forgetting and the rationality of the parameters.

The main contributions can be summarized as follows.

- We propose a novel fine-tuning method OLOR, which cooperates with optimizers to solve the knowledge forgetting issue, thereby improving fine-tuning performance.
- The designed weight rollback avoids delay defects by incorporating the current gradient into the penalty term, thereby correcting the penalty target and smoothing the review process.
- A layer-wise penalty is presented to employ penalty decay and the diversified decay rate to adjust the weight rollback levels of layers for adapting varying downstream tasks.
- The proposed method achieves state-of-the-art performance on extensive downstream tasks, including different types of image classification, different pre-trained models, and image detection and segmentation.

Related Work

Pre-training Resource

With the rapid advancement of computer vision, numerous large-scale datasets (Russakovsky et al. 2015; Schuhmann et al. 2021, 2022) and pre-trained models have emerged. These upstream pre-trained models possess rich features and hold great potential for transferability to other specific downstream tasks. ImageNet-21K (Russakovsky et al. 2015) is the most popular large-scale dataset with over 14 million images, and most networks are pre-trained on it. Recently, a groundbreaking development has taken place with the release of LAION-2B (Schuhmann et al. 2022). This dataset now reigns as the largest, comprising over 2 billion image-text pairs. Then many pre-trained models have been proposed, such as OpenClip (Radford et al. 2021), BEiT (Peng et al. 2022), MAE (He et al. 2022), and EVA (Fang et al. 2023). It is worth noting that most of these models' backbones are built upon the foundations of ViT (Dosovitskiy et al. 2020) and ConvNeXt (Liu et al. 2022).

Fine-tuning Method

The process of fine-tuning usually faces an issue known as knowledge forgetting (Toneva et al. 2018). It refers to the model's loss of pre-training learned representations during fine-tuning (Mosbach, Andriushchenko, and Klakow 2020). This leads to reduced accuracy on both the upstream and downstream tasks, as the model cannot effectively utilize its potential knowledge (De Lange et al. 2021; Vander Eeckt and Van Hamme 2023).

To solve this issue, there are currently three categories of approaches, i.e., replay methods, regularization methods, and parameter isolation methods. Replay involves periodically training on a subset of upstream task data, thereby retaining knowledge of previous tasks and balancing old and new information (Rebuffi et al.; Rolnick et al. 2019; Liu et al. 2020; Merlin et al. 2022). However, storing and managing updtream task data pose challenges in terms of efficiency, particularly in the contemporary era of massive datasets (Schuhmann et al. 2022; Li et al. 2023). Regularization-based methods employ techniques such as the fisher information matrix (Kirkpatrick et al. 2017), weight decay (Kumar et al. 2022), and L2 penalty (Xuhong, Grandvalet, and Davoine 2018) to restrict parameter updates during fine-tuning. However, these techniques may not be entirely adequate in completely preventing knowledge forgetting. Moreover, the presence of adaptive optimizers (Loshchilov and Hutter 2017; Guan 2023) can occasionally impact the direction of regularization (Xuhong, Grandvalet, and Davoine 2018). Parameter isolation methods incorporate specific branches or modules into the pre-trained network during downstream fine-tuning, aiming to achieve knowledge transfer through these new modules (Jia et al. 2022; Sohn et al. 2023; Wang et al. 2023). However, architectural modifications introduce new training parameters and intricate designs. Moreover, training tricks play a crucial role in the effectiveness of the new module, often necessitating multiple rounds of freezing and unfreezing.

To achieve a general and concise fine-tuning method to address knowledge forgetting, the proposed OLOR fine-tuning method combines weight rollback and optimizers to adjust the range of parameter updates. This allows for enhancing pre-trained model representations to improve downstream fine-tuning performance.

Method

We propose a One step Learning, One step Review (OLOR) method to reduce knowledge forgetting for fine-tuning. OLOR can be seamlessly applied to various downstream tasks among with different optimizers and models. The overall framework is illustrated in Figure 1, and detailed pipelines incorporating SGD and Adam are described in Algorithm 1 and Algorithm 2. This section introduces the delay defect of the previous regularization method, followed by detailed explanations of the OLOR method, which comprises weight rollback and layer-wise penalty.

Previous Regularization Mechanisms have a Delay Defect

The implementation of OLOR is inspired by L2 regularization and weight decay, which are popular methods used to regularize the model parameters. However, our findings indicate that their effectiveness does not align with the initial expectation. In the case of the classic SGD optimizer, L2 regularization can be regarded as equivalent to weight decay (Loshchilov and Hutter 2017), which can be defined as follows:

$$\theta_t = (1 - \lambda)\theta_{t-1} - \eta_t g_t, \tag{1}$$

where θ_t represents the model weights at iteration t, and θ_{t-1} is corresponding weights from the previous iteration. λ

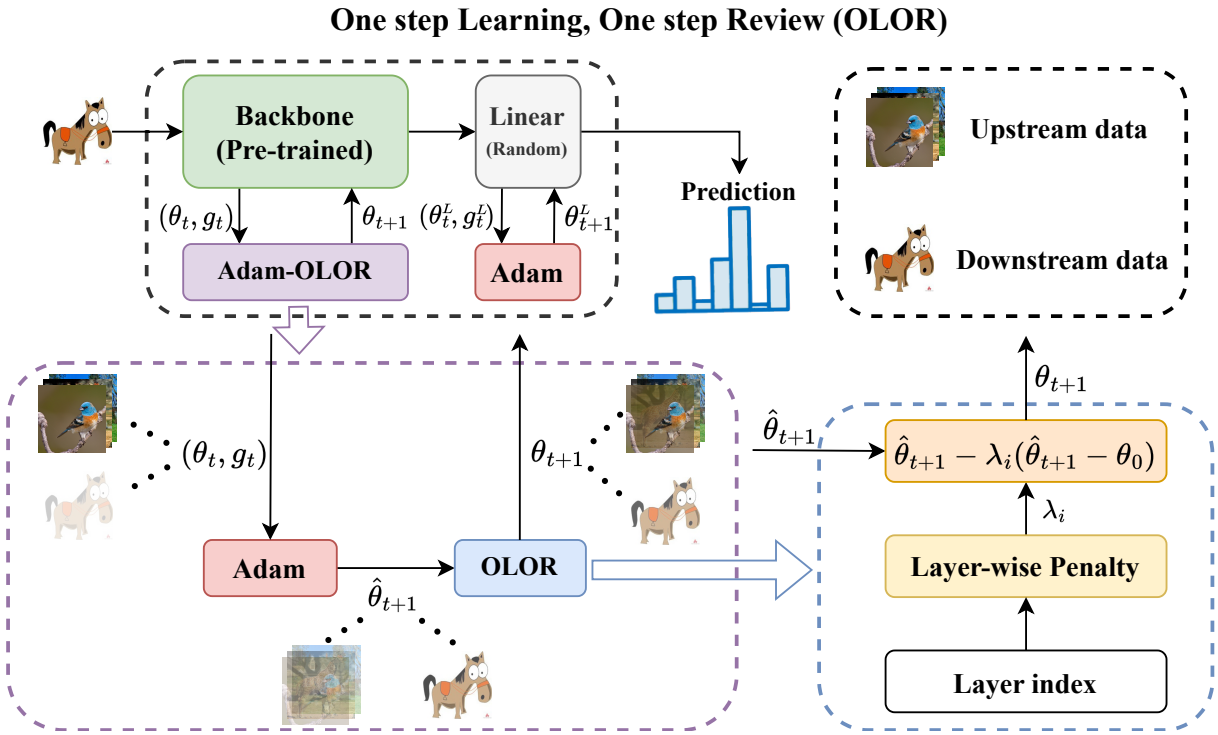

Figure 1: Overview of OLOR using Adam as optimizer. Where λ_i represents the penalty factor of i_{th} layer, θ_t and $\hat{\theta}_{t+1}$ represents the weight and the estimation of next weight (pre-weight) at timestep t, respectively. The transparency of the image indicates the knowledge forgetting level.

Algorithm 1: OLOR for SGD with Momentum

input:
$\eta \in \mathbb{R}$: Initial learning rate, $\beta \in [0, 1)$: momentum factor, θ_0: pre-trained weight, $\iota_1, \iota_2 \in [0, 1], \iota_1 \geq \iota_2$: max and min level of weight rollback respectively, $\gamma \in \mathbb{R}$: weight rollback power
initialize:
$t \leftarrow 0$: time step, $m_0 \leftarrow 0$: initial moment vector, $d_0 \leftarrow 0$: initial discrepancy value, $\lambda_i \leftarrow f(\lambda, i, n, \iota_1, \iota_2)/\eta$: calculate penalty factor λ_i through $\lambda_i = f(\lambda, i, n, \iota_1, \iota_2) = \iota_2 + (1 - \frac{i}{n})^{\gamma}(\iota_1 - \iota_2)$, then scale it by dividing η to eliminate the scale issue.
repeat
  $t \leftarrow t + 1$
  $\eta_t \leftarrow$LRScheduler(η_{t-1}) (Calculate η_t at timestep t)
  $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get batch gradient at timestep t)
  $m_t \leftarrow \beta m_{t-1} + (1 - \beta) g_t$ (Compute momentum)
  $\theta_t \leftarrow \theta_{t-1} - \eta_t \lambda_i d_{t-1} - (1 - \eta_t \lambda_i)\eta_t m_t$(Update weight)
  $d_t \leftarrow (1 - \eta_t \lambda_i)(d_{t-1} - \eta_t m_t)$ (Update discrepancy)
until Stopping condition is met
return Parameters θ_t

Algorithm 2: OLOR for Adam

input:
$\eta \in \mathbb{R}$: Initial learning rate, $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates, ϵ: bias, θ_0: pre-trained weight, $\iota_1, \iota_2 \in [0, 1], \iota_1 \geq \iota_2$: max and min level of weight rollback respectively, $\gamma \in \mathbb{R}$: weight rollback power
initialize:
$t \leftarrow 0$: time step, $m_0 \leftarrow 0$: initial first moment vector, $v_0 \leftarrow 0$: initial second moment vector, $d_0 \leftarrow 0$: initial discrepancy value, $\lambda_i \leftarrow f(\lambda, i, n, \iota_1, \iota_2)/\eta$: calculate penalty factor λ_i through $\lambda_i = f(\lambda, i, n, \iota_1, \iota_2) = \iota_2 + (1 - \frac{i}{n})^{\gamma}(\iota_1 - \iota_2)$, then scale it by dividing η to eliminate the scale issue.
repeat
  $t \leftarrow t + 1$
  $\eta_t \leftarrow$LR_Scheduler(η_{t-1}) (Calculate η_t at timestep t)
  $g_t \leftarrow \nabla f_t(\theta_{t-1})$ (Get batch gradient at timestep t)
  $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$ (Update first moment vector)
  $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$(Update second moment vector)
  $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$
  $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$
  $\theta_t \leftarrow \theta_{t-1} - \eta_t \lambda_i d_{t-1} - \frac{(1 - \eta_t \lambda_i)\eta_t \hat{m}_t}{(\sqrt{\hat{v}_t} + \epsilon)}$ (Update weight)
  $d_t \leftarrow (1 - \eta_t \lambda_i)(d_{t-1} - \frac{\eta_t \hat{m}_t}{(\sqrt{\hat{v}_t} + \epsilon)})$ (Update discrepancy)
until *stopping criterion is met*
return optimized parameters $\boldsymbol{\theta}_t$

is the regularization factor (weight decay strength). η_t is the learning rate at iteration t. g_t is the batch gradient computed from the loss function at iteration t. Weight decay penalizes the weights obtained from the previous iteration by pushing them toward 0. However, in practice, $\lim_{\lambda \to 1} \theta_t = -\eta_t g_t$, the weights tend to be pushed towards the negative value of the current gradient instead of 0. This behavior may be different from the initial expectation. Furthermore, applying weight decay can actually increase the current weight compared to not applying it. This can be seen in the following

inequality:

$$(\theta_{t-1} - \eta_t g_t - \lambda\theta_{t-1})^2 > (\theta_{t-1} - \eta_t g_t)^2, \tag{2}$$

simplified as:

$$\begin{cases} \eta g_t < (1 - \frac{\lambda}{2})\theta_{t-1}, & \text{if } \theta_{t-1} < 0, \\ \eta g_t > (1 - \frac{\lambda}{2})\theta_{t-1}, & \text{if } \theta_{t-1} > 0. \end{cases}$$

If η, g_t, λ, and θ_{t-1} are in above conditions, using weight decay will drive the current weight away from 0, which is opposite to its target. Similarly, this issue with the decay effect also exists in other regularization mechanisms such as L1 regularization, L2-SP, and similar methods.

Weight Rollback

The proposed weight rollback is a real-time regularization method that closely follows each weight update step. It aims to bring the current model weights closer to the pre-trained weights to perform knowledge reviewing. Specifically, the first step is to calculate the pre-weight θ_{pre} by gradient:

$$\theta_{pre} = \theta_{t-1} - \eta_t g_t, \tag{3}$$

where θ_{t-1} represents the model weights from the previous time step, η_t is the learning rate at the current time step, and g_t denotes the gradient. Subsequently, the discrepancy Δd between θ_{pre} and the pre-trained weight θ_0 is computed as:

$$\Delta d = \theta_{pre} - \theta_0. \tag{4}$$

Finally, the weight update process incorporates Δd, resulting in the adjusted model weights θ_t:

$$\theta_t = \theta_{t-1} - \eta_t g_t - \lambda\Delta d. \tag{5}$$

By substituting Eq. 3 and Eq. 4 into Eq. 5, we obtain:

$$\theta_t = (1 - \lambda)(\theta_{t-1} - \eta_t g_t) + \lambda\theta_0. \tag{6}$$

This Eq. 6 ensures that $\lim_{\lambda\to 1} \theta_t = \theta_0$, which aligns with our expectation and prevents abnormal scenarios. In addition, as the gradient g_t is also subject to a penalty, this process may potentially help to mitigate gradient explosions.

In summary, the weight rollback technique moderates the deviation between θ_t and θ_0 at each step, thereby alleviating overfitting to the current task and knowledge forgetting to the previous task.

Layer-wise Penalty

Penalty Decay. For deep learning neural networks, each layer can be conceptualized as a function that processes its input. Given a layer index i, this process can be described as follows:

$$x_{i+1} = f_i(x_i^*), \tag{7}$$

where the f_i represents the i_{th} layer. Let x_i^u denotes the input of f_i in upstream tasks with a distribution of $q_i(x_i^u)$, and x_i^d denotes the input of f_i in downstream tasks with a distribution of $p_i(x_i^d)$. Since $q_i(x_i^u)$ are always different from $p_i(x_i^d)$, we first unfreeze all layers to secure f_i will have sufficient update to handle such gap better.

In the study of image feature extraction, a prevailing understanding is that shallow layers are primarily responsible for capturing superficial features (Lin et al. 2017) such as color, texture, and shape. In contrast, deeper layers focus on extracting more profound features like semantic information. This implies that shallow layers are closely linked to the distribution of the data, whereas deep layers are more aligned with task-specific objectives. A foundational assumption underlying transfer learning is that $q_i(x_i^u)$ bears a degree of similarity to $p_i(x_i^d)$. Consequently, shallow layers tend to exhibit similarities in both pre-training and fine-tuning stages. Additionally, shallow layers require fewer updates compared to their deeper counterparts.

Table 1: Details of the Fine-tuning datasets.

Dataset	Images	Categories	Type
CIFAR-100	60,000	100	General
SVHN	600,000	10	General
CUB-200	11,788	200	Fine-grained
Stanford Cars	16,185	196	Fine-grained
Places-LT	62,500	365	Long-tailed
IP102	75,222	102	Long-tailed
OfficeHome	15,500	4×65	Cross-domain
PACS	9,991	4×7	Cross-domain
COCO2017	163,957	80	Detection
ADE20K	27,574	3688	Segmentation

Based on these observations, we propose a layer-wise penalty decay mechanism for weight rollback. This approach gradually reduces the rollback level as the layer depth increases. This strategy encourages shallow layers to extract more general features in downstream tasks while preserving the overall model capacity. For any layer at index i, the penalty factor λ_i is computed using the following formula:

$$\lambda_i = \iota_2 + (1 - \frac{i}{n})(\iota_1 - \iota_2), \tag{8}$$

where n represents the total number of layers in the pre-trained model, ι_1 and ι_2 denote the maximum and minimum rollback levels, respectively.

Diversified Decay Rate. Across various downstream tasks, the target objectives often exhibit varying degrees of dissimilarity from those of the upstream task. To accommodate this variability, we propose adjusting the rate of penalty decay between layers by introducing a power exponent γ to the weight rollback value. Mathematically, this adjustment can be expressed as:

$$1 - \frac{i}{n} \longrightarrow (1 - \frac{i}{n})^\gamma. \tag{9}$$

This dynamic adjustment helps mitigate the bias stemming from a fixed rate decay of the similarities between $q_i(x_i^u)$ and $p_i(x_i^d)$ across different layer indices i. Consequently, the penalty decay becomes more adaptable and versatile, catering to a spectrum of requirements dictated by the various downstream tasks.

Table 2: Comparison of fine-tuning results on various types of classification datasets (general, fine-grained, long-tailed, cross-domain).

	General (ID)		Fine-Grained (ID)		Long-Tailed (OOD)		Cross-Domain (OOD)	
Method	Cifar-100	SVHN	CUB-200	StanfordCars	Places-LT	IP102	OfficeHome	PACS
ViT-B Backbone								
Linear	72.50	58.79	75.01	38.03	31.95	64.93	79.96	71.88
Full	87.76	97.27	81.34	75.55	31.59	74.09	84.39	87.79
L2-SP	88.17	97.12	81.65	75.55	31.22	73.75	84.74	87.74
VPT	91.49	94.37	81.86	58.24	37.02	70.41	86.48	77.44
OLOR-Adam (ours)	**92.89**	**97.35**	**84.84**	**82.02**	**38.07**	**75.34**	**89.05**	**94.38**
ConvNeXt-B Backbone								
Linear	81.70	69.21	87.85	50.21	36.41	70.77	92.40	93.46
Full	92.72	96.97	88.59	88.67	38.61	75.01	91.78	95.51
L2-SP	92.84	97.01	88.82	88.83	38.52	75.20	90.61	95.90
VPT	88.71	81.58	87.88	51.58	36.32	71.22	92.31	93.75
OLOR-SGD (ours)	**92.86**	**97.12**	**89.47**	**88.99**	**39.36**	**75.44**	**92.59**	**96.63**

Experiments

Experiment Configuration

Pre-trained Backbones. The experiments employ CNN-based ConvNeXt (Liu et al. 2022) and Transformer-based Vision Transformers (ViT) (Dosovitskiy et al. 2020) as backbones. For both types of models, pre-trained weights from ImageNet-1K (MAE) (Deng et al. 2009), ImageNet-21K (supervised) (Russakovsky et al. 2015) and LAION-2B (CLIP) (Schuhmann et al. 2022) datasets are utilized, where the weights from ImageNet-21K undergoes supervised pre-training, and the others are based on self-supervised pre-training diagram.

Downstream Tasks. We experiment on ten popular visual task datasets, i.e., CIFAR-100 (Krizhevsky, Hinton et al. 2009), SVHN (Netzer et al. 2011), CUB-200 (Wah et al. 2011), Stanford Cars (Krause et al. 2013), Places-LT (Zhou et al. 2014), IP102 (Patterson et al. 2014), OfficeHome (Venkateswara et al. 2017), and PACS (Li et al. 2017), covering general classification, fine-grained classification, long-tailed classification, cross-domain classification, object detection, semantic segmentation, and instance segmentation. More details are listed in Table 1.

Baselines. To ensure a comprehensive comparison, we select the state-of-the-art and classic methods as our baselines. These encompass Full Fine-tuning (Full), Linear Probing (Linear) (Zhang, Isola, and Efros 2017), L2-SP (Xuhong, Grandvalet, and Davoine 2018), and VPT (Jia et al. 2022). Following prior works (Carion et al. 2020), CNN-based Backbones are usually combined with the SGD optimizer, while Transformer-based Backbones are paired with the Adam optimizer.

Implementation Details. The input image size is set at 224×224. The batch size varies depending on the freezing strategy. Specifically, 128, 256 and 512 are chosen for full unfreezing, parameter isolated, and full freezing based methods, respectively. Regarding the learning rate, for ConvNeXt backbones, we employ the SGD optimizer with a momentum of 0.9. The learning rates differ based on the freezing strategy. In detail, 1e-2, 2e-2 and 4e-2 for full unfreezing, parameter isolated, and full freezing based methods, respectively. For ViT backbones, we use the Adam optimizer with a momentum of (0.9, 0.999). The learning rates for ViT backbones also vary according to the freezing strategy, i.e., 1e-4 for full unfreezing, 2e-4 for partial unfreezing, and 4e-4 for full freezing. We train on cross-domain datasets for 30 epochs, while for other datasets, we train for 50 epochs. The experiments are performed on two A5000 GPU with 24 GB memory and Ubuntu 20.04 operating system. Python 3.8.3 serves as the programming language, while PyTorch 2.0.0 framework is employed. In addition, the source code is openly available on GitHub.

Main Results

Results on classification tasks. To verify the wide adaptability of OLOR on various types of datasets, we conduct a comprehensive comparison with other state-of-the-art fine-tuning methods. We evaluate these methods on 10 popular classification datasets, each showcasing a range of data distributions and characteristics. In addition, the Backbone in the experiment covers ViT-B and ConvNeXt-B, corresponding to Adam and SGD optimizers, respectively.

The experiment results are listed in Table 2. It can be observed that our OLOR achieves a new state-of-the-art on all datasets. Notably, in in-distribution (ID) datasets, OLOR-Adam surpasses the previously leading L2-SP method by an impressive margin of 6.47% in accuracy. Moreover, when confronted with two more challenging out-of-distribution (OOD) datasets, OLOR-Adam achieves accuracy improvements of 2.57% and 7.38%, respectively, outperforming the optimal methods.

Since the pre-trained ConvNeXt model is more stable than the ViT structure, there is not much difference between different methods in fine-tuning. However, our OLOR-SGD still consistently improves fine-tuning accuracy across all

Table 3: Results of object detection and instance segmentation using the ConvNeXt-B as backbone.

Method	Model	Dataset	$Bbox_m$	$Segm_m$
Full	Mask R-CNN	COCO2017	40.20	36.00
OLOR	Mask R-CNN	COCO2017	**41.10**	**36.90**

Table 4: Results of semantic segmentation using the ViT-B as backbone.

Method	Model	Dataset	IOU_m
Full	UperNet	ADE20K	43.65
OLOR	UperNet	ADE20K	**44.62**

datasets. These results demonstrate the robustness and effectiveness of the proposed OLOR in various tasks.

Results on detection and segmentation tasks. Due to the complexity of detection and segmentation tasks, most existing fine-tuning methods struggle with applicability and validation. However, integrated with the optimizer, our OLOR approach can easily be applied to these tasks. Table 3 shows the results of object detection and instance segmentation on the COCO2017 dataset, while Table 4 showcases the performance of semantic segmentation on the ADE20K dataset. OLOR consistently outperforms the Baseline by approximately 1% in all metrics, demonstrating its versatility and effectiveness in more complex detection and segmentation tasks.

Results of using different pre-trained models. Considering that the performances of different fine-tuning methods may vary when using different pre-trained models, we further conduct experiments to explore and compare. The pre-trained ViT-B model weights are obtained from ImageNet-21K (supervised), LAION-2B (CLIP), and ImageNet-1K (MAE). The fine-tuning experiments are based on the challenging PACS dataset.

As listed in Table 5, our OLOR consistently achieves state-of-the-art results across all pre-trained models. Specifically, OLOR surpasses other leading methods by 5.08%, 0.64%, and 3.47% when using Supervised, CLIP, and MAE, respectively. While other methods struggle to adapt to all pre-trained models simultaneously, our OLOR demonstrates potential across all pre-trained models.

Table 5: Results of using different pre-trained models on the PACS dataset.

Method	Supervised	OpenCLIP	MAE
Linear	71.88	95.61	36.72
Full	87.79	47.17	84.18
L2-SP	87.74	45.56	85.79
VPT	76.76	97.46	50.54
OLOR (ours)	**92.87**	**98.10**	**89.26**

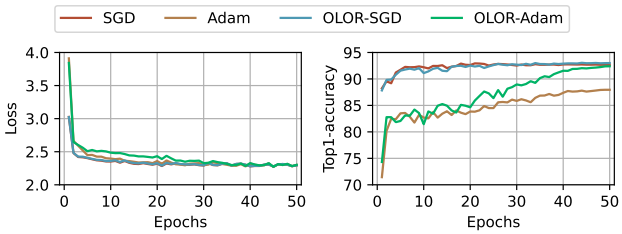

Figure 2: Train loss and valid top1 accuracy on CIfar-100, using ViT-B with Adam and ConvNext-B with SGD.

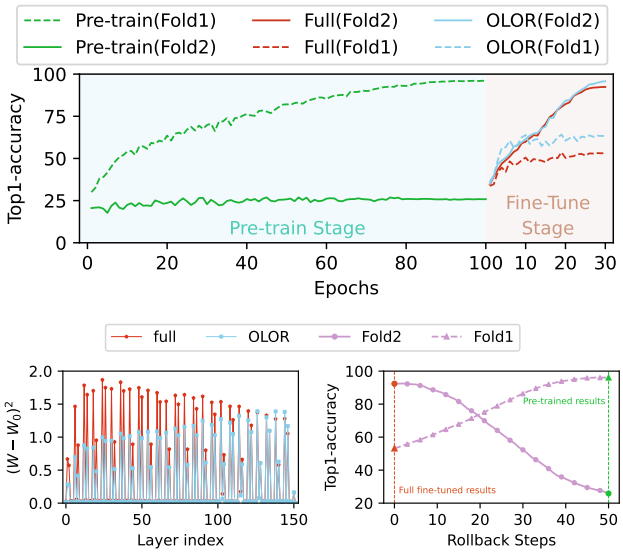

Figure 3: Knowledge forgetting test on PACS. Fold 1 as train set and fold 2 as valid set during pre-training, splits during fine-tuning is opposite to pre-training.

Summary of main results. In summary, the above experiments show that OLOR achieves SOTA when applied to multiple downstream tasks, utilizing diverse pre-trained backbones. These results demonstrate the generalizability and effectiveness of the OLOR fine-tuning method.

Analysis and Discussion

Compatibility Analysis. As shown in Figure 2, adopting weight rollback in different types of models and optimizers generally improves the performance. Due to the restriction on parameters, OLOR leads to slower loss converging speed at first, but ultimately becomes competitive with the full method. According to the validation results, OLOR potentially helps reduce knowledge forgetting, resulting in far superior top1 accuracy, especially when cooperating with Adam applied in Vision Transformers.

Knowledge Forgetting Test. To assess potential knowledge forgetting, we conduct a study on the PACS dataset using ViT-B and Adam. Firstly, split the dataset into two folds, the first fold contains data from three domains, cartoon, photo and sketch respectively, denote as $\mathcal{D}_1$, the second fold contains data from art painting domain, denote as $\mathcal{D}_2$. For training stage, we first pre-train a model using $\mathcal{D}_1$

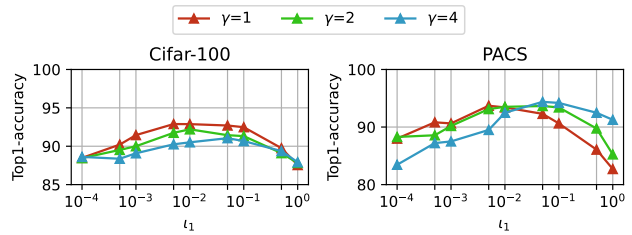

Figure 4: Hyper-parameters exploring experiments on Cifar-100(left) and PACS(right), both using ViT-B with Adam.

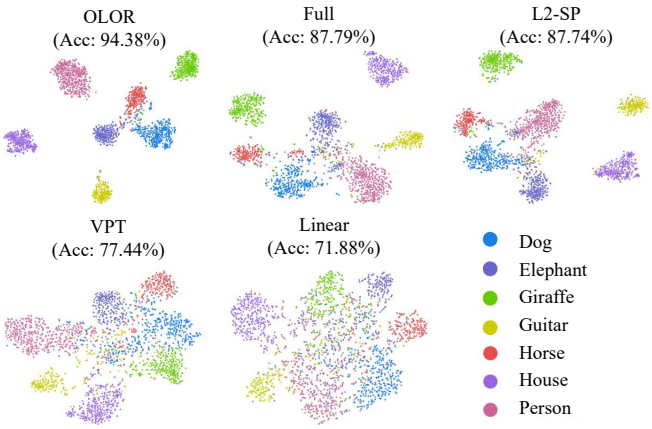

Figure 5: Feature visualization on PACS test set. We use features extracted by backbone to perform t-SNE visualization, and the Top1-accuracy are reported additionally.

as train set and $\mathcal{D}_2$ as valid set for 100 epochs, then fine-tune the model using $\mathcal{D}_2$ as train set and $\mathcal{D}_1$ as valid set for 30 epochs through Full and OLOR methods, the discrepancy between fine-tuned weight θ and pre-trained weight θ_0 using different methods are recorded. Additionally, we perform zero-shot reviewing, rolling back full fine-tuned weights to pre-trained weights in 50 steps. Figure 3 reports the results, weight discrepancy is generally much smaller using OLOR, when setting max rollback level ι_1 to 0.01, rollback power γ to 1, OLOR not only performs well in knowledge reviewing, but also benefits for current learning. And the zero-shot reviewing result shows weight rollback itself is indeed a helpful method for just reviewing.

Hyper-parameter Exploration. We conduct experiments on Cifar-100(ID) and PACS(OOD) to study the appropriate hyper-parameters for different types of tasks. Deep layers usually require significant updates to effectively extract features related to the downstream task, thus we set the min rollback level ι_2 to 0 by default to simplify hyper-parameter settings, for max rollback level ι_1, we search from $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1\}$, for weight rollback power γ, we search from $\{1, 2, 4\}$. Figure 4 shows the findings. We suggest applying small power if the task target of the fine-tuning stage is similar to the pre-training stage, and large max rollback level if the data distribution of downstream task is similar to upstream task.

Table 6: Hyper-parameter configuration of OLOR for different downstream tasks.

Datasets	ViT-Based			CNN-based		
	ι_1	ι_2	γ	ι_1	ι_2	γ
Cifar-100	5e-3	0	2	5e-3	0	2
SVHN	5e-3	0	2	1e-4	0	2
CUB-200	5e-2	0	2	1e-2	0	2
StanfordCars	1e-2	0	4	1e-4	0	2
Places-LT	1e-1	0	4	1e-2	0	4
IP102	1e-1	0	1	5e-3	0	1
OfficeHome	1e-2	0	1	1	0	1
PACS	1e-1	0	4	5e-2	0	4
COCO2017	-	-	-	1e-2	0	2
ADE20K	1e-4	0	1	-	-	-

Table 7: Hyper-parameter configuration of OLOR for different pre-trained models.

Pre-trained Method	ι_1	ι_2	γ
Supervised	1e-2	0	2
OpenCLIP	1e-2	0	2
MAE	1e-2	0	2

Feature Visualization. We visualized the feature distributions for all methods on PACS test set through t-SNE to evaluate the quality of the extracted features. Experiments are based on ViT-B and Adam. As shown in Figure 5, compared with previous methods, OLOR generally separates the representation vectors of different classes much better, demonstrating superior ability on representation.

Conclusions

In this paper, we propose a novel fine-tuning method named OLOR to solve the challenge of knowledge forgetting in neural networks. OLOR encompasses weight rollback and layer-wise penalty. OLOR incorporates the weight rollback term into the weight update term at each step, and can be implemented in popular optimizers. This operation allows the model to gradually approach the pre-trained weights while learning the downstream task, making the weights of the upstream and downstream models more similar. In addition, the layer-wise penalty employs penalty decay and the diversified decay rate to adjust the weight rollback levels of layers for adapting varying downstream tasks. Our OLOR achieves state-of-the-art performance on extensive downstream tasks. Validation experiments and ablation analysis demonstrate the effectiveness of the proposed method.

Additional Implementation Details

In the Main Results section, when conducting experiments on various downstream tasks, OLOR utilizes the hyper-parameter configurations listed in Table 6. For experiments involving different pre-trained models, the hyper-parameter configurations for OLOR are listed in Table 7.

References

Bao, H.; Dong, L.; Piao, S.; and Wei, F. 2021. Beit: Bert pre-training of image transformers. *ArXiv Preprint arXiv:2106.08254*.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.

De Lange, M.; Aljundi, R.; Masana, M.; Parisot, S.; Jia, X.; Leonardis, A.; Slabaugh, G.; and Tuytelaars, T. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7): 3366–3385.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. Ieee.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv Preprint arXiv:2010.11929*.

Fang, Y.; Sun, Q.; Wang, X.; Huang, T.; Wang, X.; and Cao, Y. 2023. Eva-02: A visual representation for neon genesis. *arXiv preprint arXiv:2303.11331*.

Guan, L. 2023. Weight Prediction Boosts the Convergence of AdamW. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 329–340. Springer.

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16000–16009.

Jia, M.; Tang, L.; Chen, B.-C.; Cardie, C.; Belongie, S.; Hariharan, B.; and Lim, S.-N. 2022. Visual prompt tuning. In *European Conference on Computer Vision*, 709–727. Springer.

Keskar, N. S.; and Socher, R. 2017. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

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

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kumar, A.; Shen, R.; Bubeck, S.; and Gunasekar, S. 2022. How to fine-tune vision models with sgd. *arXiv preprint arXiv:2211.09359*.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2017. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, 5542–5550.

Li, Y.; Zhang, K.; Liang, J.; Cao, J.; Liu, C.; Gong, R.; Zhang, Y.; Tang, H.; Liu, Y.; Demandolx, D.; et al. 2023. Lsdir: A large scale dataset for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1775–1787.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2117–2125.

Liu, X.; Wu, C.; Menta, M.; Herranz, L.; Raducanu, B.; Bagdanov, A. D.; Jui, S.; and de Weijer, J. v. 2020. Generative feature replay for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 226–227.

Liu, Z.; Mao, H.; Wu, C.-Y.; Feichtenhofer, C.; Darrell, T.; and Xie, S. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11976–11986.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Merlin, G.; Lomonaco, V.; Cossu, A.; Carta, A.; and Bacciu, D. 2022. Practical recommendations for replay-based continual learning methods. In *International Conference on Image Analysis and Processing*, 548–559. Springer.

Mosbach, M.; Andriushchenko, M.; and Klakow, D. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning.

Patterson, G.; Xu, C.; Su, H.; and Hays, J. 2014. The sun attribute database: Beyond categories for deeper scene understanding. *International Journal of Computer Vision*, 108: 59–81.

Peng, Z.; Dong, L.; Bao, H.; Ye, Q.; and Wei, F. 2022. Beit v2: Masked image modeling with vector-quantized visual tokenizers. *arXiv preprint arXiv:2208.06366*.

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

Rebuffi, S.; Kolesnikov, A.; Sperl, G.; Lampert, C.; and icarl. ???? Incremental classifier and representation learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 5533–5542.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115: 211–252.

Schuhmann, C.; Beaumont, R.; Vencu, R.; Gordon, C.; Wightman, R.; Cherti, M.; Coombes, T.; Katta, A.; Mullis, C.; Wortsman, M.; et al. 2022. Laion-5B: An open large-scale dataset for training next generation image-text models. *ArXiv Preprint arXiv:2210.08402*.

Schuhmann, C.; Vencu, R.; Beaumont, R.; Kaczmarczyk, R.; Mullis, C.; Katta, A.; Coombes, T.; Jitsev, J.; and Komatsuzaki, A. 2021. Laion-400M: Open dataset of clip-filtered 400 million image-text pairs. *ArXiv Preprint arXiv:2111.02114*.

Shen, Z.; Liu, Z.; Qin, J.; Savvides, M.; and Cheng, K.-T. 2021. Partial is better than all: revisiting fine-tuning strategy for few-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 9594–9602.

Sohn, K.; Chang, H.; Lezama, J.; Polania, L.; Zhang, H.; Hao, Y.; Essa, I.; and Jiang, L. 2023. Visual prompt tuning for generative transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19840–19851.

Toneva, M.; Sordoni, A.; Combes, R. T. d.; Trischler, A.; Bengio, Y.; and Gordon, G. J. 2018. An empirical study of example forgetting during deep neural network learning. *ArXiv Preprint ArXiv:1812.05159*.

Vander Eeckt, S.; and Van Hamme, H. 2023. Using adapters to overcome catastrophic forgetting in end-to-end automatic speech recognition. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5. IEEE.

Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5018–5027.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset.

Wang, R.; Zheng, H.; Duan, X.; Liu, J.; Lu, Y.; Wang, T.; Xu, S.; and Zhang, B. 2023. Few-Shot Learning with Visual Distribution Calibration and Cross-Modal Distribution Alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 23445–23454.

Wu, W.; Sun, Z.; and Ouyang, W. 2023. Revisiting classifier: Transferring vision-language models for video recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2847–2855.

Xuhong, L.; Grandvalet, Y.; and Davoine, F. 2018. Explicit inductive bias for transfer learning with convolutional networks. In *International Conference on Machine Learning*, 2825–2834. PMLR.

Zhang, R.; Isola, P.; and Efros, A. A. 2017. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1058–1067.

Zhou, B.; Lapedriza, A.; Xiao, J.; Torralba, A.; and Oliva, A. 2014. Learning deep features for scene recognition using places database. *Advances in neural information processing systems*, 27.