Initialization Matters for Adversarial Transfer Learning

Supplementary Material

A. Experimental Details

Pretrained Models. In this paper, we use ViT-B/16 [**?**] and Swin-B [**?**] for all experiments. For standard pretrained models, we use the model in Torchvision for ImageNet-1k ViT, while obtaining the weights for CLIP ViT from hugging face [1]. Additionally, we download the weights for the ImageNet-1k Swin transformer from the official implementation[2].

The adversarially robust models are pretrained on ImageNet-1k using PGD-3 with $\varepsilon = 4/255$ and a step size of $2\varepsilon/3$ following [**?** **?**]. All weights for the adversarially robust pretrained models can be found at ARES 2.0 benchmark[3].

Datasets Split. For CIFAR10 and CIFAR100 [**?**], we follow the official train-validation-test splits. For Caltech256 [**?**], we randomly select 14,649 images (57 images per class) for training and 771 images (3 images per class) for validation, leaving the remaining images for the test set. For CUB200 [**?**], we adopt the official train-test split and randomly select 600 images from the train set (3 images per class) as the validation set. For Stanford Dogs [**?**], we also follow the official train-test split and randomly pick 1200 images from the train set (10 images per class) as the validation set.

PEFT Hyperparameters. In this paper, we use six fine-tuning methods including Full Finetune, Adapter, LoRA, Bias, VPT, and Linear. For Full Finetune, Bias, and Linear, no extra hyperparameters are required as they fine-tune either part or the entirety of the model parameters.

For Adapter, we adopt the architecture proposed in [**?**], which involves appending the adapter module after the MLP block at each layer. Additionally, we apply a residue connection for the adapter module. Throughout the paper, we maintain a reduction factor of 8. When implementing RoLI, we initialize the downsample and upsample layers within the adapter module to zero.

We only apply the LoRA branch on the query and value projection in the self-attention block. Following [**?**], with pretrained weights W_0 and input x, the LoRA branch can be formulated as Eq. (1)

$$h = W_0 x + \frac{\alpha}{r} BAx \tag{1}$$

where A and B represent low-rank decomposition matrices with a rank of r. α demotes a scaling constant. We set both the rank r and scaling factor α to 16 across all datasets. For random linear initialization (RanLI), we adopt the initialization proposed in [**?**], applying random Gaussian initialization for A and zero initialization for B. For robust linear initialization (RoLI), we zero matrices A and B.

For VPT, we adopt VPT-Deep in [**?**]. We set the number of tokens as 10 across all datasets. We apply uniform initialization for prompts in RanLI. We do not apply RoLI to VPT because even when prompts are zeroed, the resulting output remains distinct from the output obtained without prompts. This difference can be attributed to the influence of the softmax operation within the attention module.

Optimizatation Hyperparameters. We finetune the model for 20 epochs on CIFAR10 and CIFAR100, 40 epochs on Caltech256 and Standford Dogs, 60 epochs on CUB200. Furthermore, we conduct a grid search to determine the optimal learning rate and weight decay based on validation performance. The weight decay search range is consistent at {0.01, 0.001, 0.0001, 0} across all fine-tuning methods and datasets.

The learning rate is set as $base\_lr \times b/256$. Regarding the base learning rates, for RanLI, we explore {0.005, 0.001, 0.0001, 0.0005} for Full Finetune and Bias, {1.0, 0.5, 0.1, 0.05} for Linear, {0.5, 0.1, 0.05, 0.01} for VPT, and {0.05, 0.01, 0.005, 0.001} for Adapter and LoRA. For RoLI, we expand the search range to include values an order of magnitude lower, such as {0.005, 0.001, 0.0001, 0.0005, 0.00001, 0.00005} for Full Finetune.

We summarize the optimal hyperparameters combination in Tab. 2.

Transferred Accuracy and Transferred Robustness ?? demonstrates that transferred robustness is correlated with transferred accuracy. We train the model using various hyperparameters and pair the accuracy in standard linear probing with the robustness in adversarial linear probing using the same hyperparameters. For example, following the notation introduced in **??**, a pair obtained through finetuning is $[acc(FT_{std})_i, rob(FT_{adv})_i]$, while a pair resulting from linear probing is $[acc(LP_{std})_j, rob(LP_{adv})_j]$.

Following this, we calculated the transferred accuracy and robustness between any two pairs—comparing one from full finetuning and another from linear probing:

$$(\text{acc}^T_{std})_{ij} = \frac{\text{acc}(LP_{std})_j - \text{acc}(FT_{std})_i}{\text{acc}(FT_{std})_i} \tag{2}$$

[1] LAION-2B CLIP weights checkpoint
[2] Swin Transformer checkpoint
[3] Adversarially robust pretrained models

Initialization	RegLI	StdLI	RoLI
Accuracy	92.08	92.75	91.55

Table 1. Accuracy of different initialization methods.

$$(\text{rob}^T_{adv})_{ij} = \frac{\text{rob}(LP_{adv})_j - \text{rob}(FT_{adv})_i}{\text{rob}(FT_{adv})_i} \quad (3)$$

Where $(\text{acc}^T_{std})_{ij}$ and $(\text{rob}^T_{adv})_{ij}$ are transferred accuracy and transferred robustness for pairs i and j. For example, on CIFAR10, where we used 5 pairs for full fine-tuning and 6 pairs for linear probing, our plot on **??** will display 30 data points representing these comparisons.

Speed Analysis In **??**, we analyze the trade-off between robustness and speed. We use a single NVIDIA RTX A6000 to conduct the experiments. For RanLI, we use the same hyperparameter search strategy and training epochs. The only difference is that RoLI applies adversarial training while RanLI does not. For RegLI, we implement logistic regression using the sklearn [**?**] library. Additionally, we also summarize the performance for these initialization methods without further adversarial finetuning in Tab. 1.

B. More Ablation Studies

RoLI Mitigates Overfitting. Overfitting is a general phenomenon for adversarial training, as highlighted in prior research [**?** **?**]. Although intuitively PEFT methods could mitigate overfitting as they only update a small number of parameters, we observe that they still suffer from overfitting. Taking adversarial Adapter on Stanford Dogs as an example in Fig. 1, RanLI suffers from overfitting, while RoLI effectively mitigates this phenomenon. Specifically, adversarial linear probing (RoLI) avoids overfitting possibly because it does not change the features during finetuning. In addition, when we initialize adversarial Adapter with RoLI, we can use a smaller learning rate during finetuning, resulting in a further decrease in loss without overfitting.

Image Gradient Visualization. In this section, we present visualizations of the loss gradient with respect to the input image for different PEFT methods using either an adversarially robust or non-robust pretrained model on the Caltech256 dataset. Fig. 2 illustrates semantically structured gradients obtained from adversarially robust pretraining, whereas non-robust pretraining yields less semantically structured gradients, except fully finetuning. This observation aligns with our results, where fully finetuning outperforms other methods when a non-robust pretraining is applied. In contrast to fully finetuning, the image gradients from PEFT methods contain less semantic information, further validating our results.

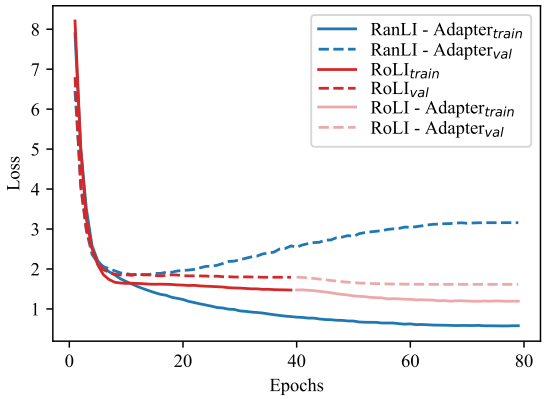

Figure 1. **RoLI mitigates overfitting.** For RoLI, the loss for the adversarial linear probing stage and Adapter finetuning stage is denoted as red and light red lines, respectively. The decreasing validation loss demonstrates that RoLI can mitigate overfitting.

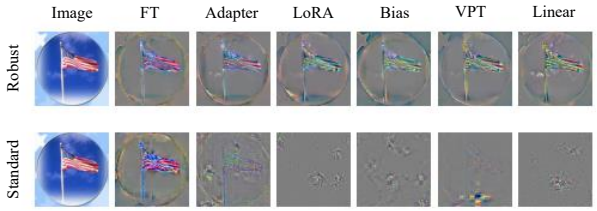

Figure 2. A robust pretrained model exhibits more semantically structured gradients compared to a standard pretrained model.

C. RoLI Pseudo Code

Algorithm 1 RoLI with adversarial finetuning

Input: Dataset D, Model θ, Parameters in linear head θ_{head}, Tunable parameters in each adversarial finetuning method θ_{ft}, Perturbation bound ε, Loss function $\mathcal{L}$
Output: Adversarially robust model θ^*
// Initialize tunable parameters in linear head.
Random initialize θ_{head}
// Robust Linear Initialization
$\hat{\theta} \leftarrow \theta_{head}, \theta_{frozen} \leftarrow \theta \setminus \theta_{head}$
$\hat{\theta} \leftarrow \min_{\hat{\theta}} \mathbb{E}_{(x,y)\sim D} \left[\max_{\|\delta\|_\infty \leq \varepsilon} \mathcal{L}(x+\delta, y; \hat{\theta} \cup \theta_{frozen})\right]$
$\theta \leftarrow \hat{\theta} \cup \theta_{frozen}$
// Adversarial FT, tunable parameters shown in Fig. 1
$\hat{\theta} \leftarrow \theta_{ft}, \theta_{frozen} \leftarrow \theta \setminus \theta_{ft}$
$\hat{\theta} \leftarrow \min_{\hat{\theta}} \mathbb{E}_{(x,y)\sim D} \left[\max_{\|\delta\|_\infty \leq \varepsilon} \mathcal{L}(x+\delta, y; \hat{\theta} \cup \theta_{frozen})\right]$
$\theta^* \leftarrow \hat{\theta} \cup \theta_{frozen}$

D. Numerical Results

For **??**, the numerical results are in **??** average column. The results for TWINS-AT [**?**] and AutoLoRa [**?**] are from their original papers. We show the numerical results for **??** in Tab. 3. We do not include the CLIP results for Swin Transformer because the CLIP pretrained weights are only available for ResNet-50 [**?**] and ViT [**?**]. Tab. 4 displays the numerical results corresponding to **??** and **??**.

Datasets	CIFAR10		CIFAR100		Caltech256		CUB200		Dogs	
Hyperparameters	Base Lr	Wd	Base Lr	Wd	Base Lr	Wd	Base Lr	Wd	Base Lr	Wd
RanLI - Full-FT	0.0005	0.01	0.0005	0.001	0.0005	0.0001	0.0005	0.001	0.0005	0.01
RanLI - Adapter	0.05	0.01	0.005	0.0001	0.005	0.0001	0.05	0.0001	0.005	0.0001
RanLI - LoRa	0.01	0.0001	0.05	0	0.001	0.0001	0.005	0.0001	0.005	0.0001
RanLI - Bias	0.005	0.001	0.005	0.001	0.001	0.01	0.005	0.001	0.001	0.0001
RanLI - VPT	0.5	0.001	0.5	0.01	0.05	0.001	0.05	0.001	0.05	0.0001
RanLI - Linear	0.05	0.01	0.1	0.001	0.5	0.01	1	0.001	1	0.01
RoLI - Full-FT	0.0001	0.01	0.0001	0.001	0.0001	0.0001	0.0001	0.01	0.00005	0.001
RoLI - Adapter	0.05	0.01	0.05	0.0001	0.0005	0	0.05	0.001	0.0001	0.0001
RoLI - LoRa	0.001	0.0001	0.001	0.0001	0.0001	0	0.005	0.0001	0.001	0.01
RoLI - Bias	0.005	0.001	0.005	0.01	0.00001	0	0.0005	0.0001	0.00005	0.0001

Table 2. Summary of the optimal hyperparameter combination across five datasets.

Model	Methods	CIFAR10						Caltech256					
		IN		CLIP		Robust IN		IN		CLIP		Robust IN	
		Clean	PGD	Clean	PGD	Clean	PGD	Clean	PGD	Clean	PGD	Clean	PGD
ViT	Full-FT	87.91	50.45	78.53	45.26	89.32	51.62	52.82	25.91	37.16	5.29	71.01	36.68
	Adapter	64.19	29.13	58.70	28.51	88.17	52.63	46.24	10.94	7.95	6.60	78.11	46.51
	LoRa	82.42	47.68	69.43	41.01	90.03	56.28	1.01	1.07	4.86	4.86	75.22	45.16
	Bias	56.30	31.19	55.85	32.06	88.29	52.94	41.07	16.07	8.59	4.95	77.09	47.19
	VPT	68.61	31.4	40.43	24.87	88.64	54.25	36.94	9.89	7.70	4.86	75.54	46.79
	Linear	52.99	0.21	74.88	0.00	84.42	35.81	53.80	0.50	26.06	0.54	79.35	48.22
Swin	Full-FT	87.54	51.79	-	-	93.02	59.12	62.71	34.77	-	-	77.35	49.13
	Adapter	70.32	32.83	-	-	90.23	55.70	41.48	10.27	-	-	80.24	52.73
	LoRa	80.55	47.67	-	-	90.02	56.88	0.29	0.28	-	-	77.02	50.25
	Bias	62.25	35.02	-	-	90.25	56.94	1.44	1.44	-	-	79.50	52.43
	VPT	66.90	35.12	-	-	90.24	57.38	45.22	18.36	-	-	79.74	53.07
	Linear	42.13	1.62	-	-	84.16	37.30	25.92	0.60	-	-	81.79	54.47

Table 3. **Numerical results for ??.** We omit the results obtained from CLIP with Swin Transformer because the pretrained model is unavailable.

Model	Methods	CIFAR10					Caltech256				
		Standard FT		Adversarial FT		Δ PGD	Standard FT		Adversarial FT		Δ PGD
		Clean	PGD	Clean	PGD		Clean	PGD	Clean	PGD	
ViT	Full-FT	97.58	9.57	89.32	51.62	42.05	80.02	6.41	71.01	36.68	30.27
	Adapter	97.65	8.15	88.17	52.63	44.48	85.59	9.65	78.11	46.51	36.86
	LoRa	97.80	21.58	90.03	56.28	34.70	84.85	6.49	75.22	45.16	38.67
	Bias	97.95	9.35	88.29	52.94	43.59	86.66	4.09	77.09	47.19	43.10
	VPT	97.51	10.22	88.64	54.25	44.03	85.27	8.96	75.54	46.79	37.83
	Linear	89.57	19.87	84.42	35.81	15.94	82.85	32.72	79.35	48.22	15.50
Swin	Full-FT	98.66	0.92	93.02	59.12	58.20	87.89	0.16	77.35	49.13	48.97
	Adapter	98.68	4.26	90.23	55.70	51.44	87.35	3.90	80.24	52.73	48.83
	LoRa	98.49	13.51	90.02	56.88	43.37	87.24	1.49	77.02	50.25	48.76
	Bias	98.80	7.06	90.25	56.94	49.88	88.49	0.79	79.50	52.43	51.64
	VPT	98.39	14.62	90.24	57.38	42.76	86.61	11.50	79.74	53.07	41.57
	Linear	90.33	21.91	84.16	37.30	15.39	85.01	47.10	81.79	54.47	7.37

Table 4. Numerical results for **??** and **??**.