Supplementary materials of Adaptive Random Feature Regularization on Fine-tuning Deep Neural Networks

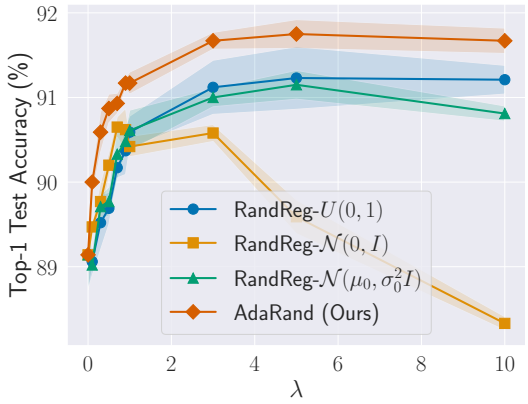

Figure I. Top-1 Accuracy on Cars (ResNet-50)

Table I. Top-1 Test Accuracy on Cars (ResNet-50)

EMA decay param. α	Test Accuracy (%)
0.0 (Only using $\hat{\mu}_k$)	$90.89^{\pm.10}$
0.1	$90.96^{\pm.20}$
0.3	$91.14^{\pm.26}$
0.5	$91.17^{\pm.13}$
0.7	$91.32^{\pm.04}$
0.9	$91.27^{\pm.12}$
0.99	$91.13^{\pm.18}$
0.999	$91.26^{\pm.19}$
1.0 (Fixed $\bar{\mu}_k$)	$90.62^{\pm.04}$

A. Effects of Hyperparameters

In the main paper, we fixed the hyperparameters of the trade-off parameter λ in Eq. (1) and decay parameter α in Eq. (14) for fair comparison. Here, we confirm the effects of varying them on the performance.

Trade-off parameter λ in Eq. (1). The trade-off parameter λ in Eq. (1) controls the regularization effect by the random reference vectors. We fixed $\lambda = 1.0$ for a fair comparison, but evaluating the sensitivity of λ is important in practice. Figure I shows the results varying λ among $\{0.0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0, 3.0, 5.0, 10.0\}$. We observed that our AdaRand stably outperformed the RandReg variants by large margins. This indicates that AdaRand is not sensitive to the choice of λ.

EMA decay parameter α in Eq. (14). As described in Sec. 5.2 in the main paper, we update the prior parameter set $\boldsymbol{\mu}$ according to the running mean vector set $\bar{\boldsymbol{\mu}}$. To compute $\bar{\boldsymbol{\mu}}$, we introduce the exponential moving average in Eq. (14) and (15) as

$$\bar{\mu}_k \leftarrow \alpha \bar{\mu}_k + (1-\alpha)\hat{\mu}_k,$$
$$\hat{\mu}_k = \tfrac{1}{B_k} \textstyle\sum_{i=1}^{B_k} g_\phi(x^i),$$

That is, the hyperparameter α controls the decay of the past information in $\bar{\boldsymbol{\mu}}$. If α is zero, then we use only batch-wise feature mean vectors for updating $\boldsymbol{\mu}$, and if α is one, then we fix $\bar{\boldsymbol{\mu}}$ in its initial state. We evaluated the effect of α on the Cars dataset. Table I shows that the performance of AdaRand has certain robustness to the choice of α as long as it uses the EMA update of $\bar{\boldsymbol{\mu}}$, i.e., $0 < \alpha < 1$, and setting α to the range of $[0.5, 0.9]$ achieves better accuracy scores than the others. This indicates that updating $\bar{\boldsymbol{\mu}}$ by EMA is important to the performance improvements, and we can slightly improve the performance by tuning α.

Table II. Fine-tuning CLIP (ViT-B/16).

Method	ImageNet	ImageNet-A	ImageNet-R
Fine-tuning	79.74	18.31	45.40
FNP	79.95	20.36	48.31
DR-Tune	31.91	1.51	4.48
RandReg-$U(0,1)$	80.08	20.27	47.01
RandReg-$\mathcal{N}(0,1)$	77.35	17.47	43.25
RandReg-$\mathcal{N}(\mu_0, \sigma_0^2)$	80.04	20.27	47.79
AdaRand (Ours)	**82.00**	**23.50**	**51.72**

Table III. Object Detection and Segmentation on VOC-2012.

Method	bbox AP	mask AP
Fine-tuning	43.5	32.2
RandReg-$U(0,1)$	42.8	31.3
RandReg-$\mathcal{N}(0,1)$	41.2	30.1
AdaRand (Ours)	**45.9**	**35.2**

B. Additional Experiments

B.1. Fine-tuning CLIP on ImageNet

We evaluate our method on large-scale target datasets. Table II shows the result of fine-tuning CLIP on ImageNet (5 epoch). Our method is effective even when the class numbers and training dataset are large, while RandReg sometimes degrades baselines. In particular, our method significantly improves the performance of ImageNet-R, indicating that our method also enhances the generalizability to real-world distribution shifts.

B.2. Application to Object Detection

We evaluate our method on non-classification tasks to show the generalizability across tasks. We tested our method on the Pascal VOC-2012 object detection and semantic segmentation with Mask-RCNN. Table III shows that our method boosts the baseline even on non-classification tasks. Note that our method can be naïvely applied to the Pascal VOC-2012 because each image has a single category object in the dataset, but it is non-trivial when input images have multiple category objects like the COCO dataset. We will explore extending our method to arbitrary tasks in future work.