Supplementary Material

PADS: Policy-Adapted Sampling for Visual Similarity Learning

Anonymous CVPR submission

Paper ID 1485

1. Additional Ablation Experiments

We now conduct further ablation experiments for different aspects of our proposed approach based on the CUB200-2011[19] dataset. Note, that like in our main paper we did not apply any learning rate scheduling for the results of our approach to establish comparable training settings.

Performance with Inception-BN: For fair comparison, we also evaluate using Inception-V1 with Batch-Normalization [4]. We follow the standard pipeline (see e.g. [11, 13]), utilizing Adam [7] with images resized and random cropped to 224x224. The learning rate is set to 10^{-5}. We retain the size of the policy network and other hyperparameters. The results on CUB200-2011[19] and CARS196[8] are listed in Table 1. On CUB200, we achieve results competitive to previous state-of-the-art methods. On CARS196, we achieve a significant boost over baseline values and competitive performance to the state-of-the-art.

Validation set $\mathcal{I}_{\text{val}}$: The validation set $\mathcal{I}_{val}$ is sampled from the training set $\mathcal{I}_{train}$, composed as either a fixed disjoint, held-back subset or repetitively re-sampled from $\mathcal{I}_{train}$ during training. Further, we can sample $\mathcal{I}_{val}$ across all classes or include entire classes. We found (Tab. 2 (d)) that sampling $\mathcal{I}_{val}$ from each class works much better than doing it per class. Further, resampling $\mathcal{I}_{val}$ provides no significant benefit at the cost of an additional hyperparameter to tune.

Composition of states s and target metric e: Choosing meaningful target metrics $e(\phi(\cdot;\zeta),\mathcal{I}_{\text{val}})$ for computing rewards r and a representative composition of the training state s increases the utility of our learned policy π_θ. To this end, Tab. 3 compares different combinations of state compositions and employed target metrics e. We observe that incorporating information about the current structure of the embedding space Φ into s, such as intra- and inter-class distances, is most crucial for effective learning and adaptation. Moreover, also incorporating performance metrics into s which directly represent the current performance of the model ϕ, e.g. Recall@1 or NMI, additional adds some useful information.

Frequency of updating π_θ: We compute the reward r for an adjustment a to $p(I_n|I_a)$ every M DML training iterations. High values of M reduce the variance of the rewards r, however, at the cost of slow policy updates which result in potentially large discrepancies to updating ϕ. Tab. 4 (a) shows that choosing M from the range $[30, 70]$ results in a good trade-off between the stability of r and the adaptation of $p(I_n|I_a)$ to ϕ. Moreover, we also show the result for setting $M=\infty$, i.e. using the initial distribution throughout training without adaptation. Fixing this distribution performs worse than the reference method Margin loss with static distance-based sampling[22]. Nevertheless, frequently adjusting $p(I_n|I_a)$ leads to significant superior performance, which indicates that our policy π_θ effectively adapts $p(I_n|I_a)$ to the training state of ϕ.

Importance of long-term information for states s: For optimal learning, s should not only contain information about the current training state of ϕ, but also about some history of the learning process. Therefore, we compose s of a set of running averages over different lengths $\mathcal{R}$ for various training state components, as discussed in the implementation details of the main paper. Tab. 4 (b) confirms the importance of long-term information for stable adaptation and learning. Moreover, we see that the set of moving averages $\mathcal{R}=\{2,8,16,32\}$ works best.

2. Curriculum Evaluations

In Fig. 1 we visually illustrate the fixed curriculum schedules which we applied for the comparison experiment in Sec. 5.3 of our main paper. We evaluated various schedules - Linear progression of sampling intervals starting at semi-hard negatives going to hard negatives, and progressively moving $\mathcal{U}$-dist[22] towards harder negatives. The schedules visualized were among the best performing ones to work for both CUB200 and CARS196 dataset.

Dataset		CUB200-2011[19]				CARS196[8]			
Approach	Dim	R@1	R@2	R@4	NMI	R@1	R@2	R@4	NMI
HTG[24]	512	59.5	71.8	81.3	-	76.5	84.7	90.4	-
HDML[25]	512	53.7	65.7	76.7	62.6	79.1	87.1	92.1	69.7
HTL[1]	512	57.1	68.8	78.7	-	81.4	88.0	92.7	-
DVML[9]	512	52.7	65.1	75.5	61.4	82.0	88.4	93.3	67.6
A-BIER[12]	512	57.5	68.7	78.3	-	82.0	89.0	93.2	-
MIC[14]	128	66.1	76.8	85.6	69.7	82.6	89.1	93.2	68.4
D&C[15]	128	65.9	76.6	84.4	69.6	84.6	90.7	94.1	70.3
Margin[22]	128	63.6	74.4	83.1	69.0	79.6	86.5	90.1	69.1
Reimpl. Margin[22], IBN	512	63.8	75.3	84.7	67.9	79.7	86.9	91.4	67.2
Ours(Margin[22] + *PADS*, IBN)	512	66.6	77.2	85.6	68.5	81.7	88.3	93.0	68.2
Significant increase in network parameter:									
HORDE[5]+Contr.[2]	512	66.3	76.7	84.7	-	83.9	90.3	94.1	-
SOFT-TRIPLE[13]	512	65.4	76.4	84.5	-	84.5	90.7	94.5	70.1
Ensemble Methods:									
Rank[20]	1536	61.3	72.7	82.7	66.1	82.1	89.3	93.7	71.8
DREML[23]	9216	63.9	75.0	83.1	67.8	**86.0**	**91.7**	**95.0**	**76.4**
ABE[6]	512	60.6	71.5	79.8	-	85.2	90.5	94.0	-

Table 1: Comparison to the state-of-the-art DML methods on CUB200-2011[19] and CARS196[8] using the Inception-BN Backbone (see e.g. [11, 13]) and embedding dimension of 512.

Validation Set:	$\mathcal{I}_{\text{val}}^{\text{By}}$	$\mathcal{I}_{\text{val}}^{\text{Per}}$	$\mathcal{I}_{\text{val}}^{\text{By, R}}$	$\mathcal{I}_{\text{val}}^{\text{Per, R}}$
Recall@1	62.6	65.7	63.0	**65.8**
NMI	67.7	69.2	67.8	**69.6**

Table 2: Composition of $\mathcal{I}_{\text{val}}$. Superscript By/Per denotes usage of entire classes/sampling across classes. R denotes re-sampling during training with best found frequency of $\frac{1}{50 \text{ epochs}}$.

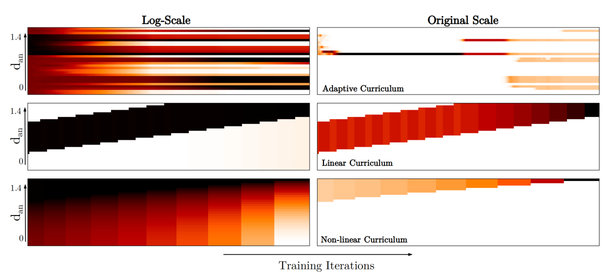

Figure 1: *Visual comparison between fixed sampling curriculums and a learned progression of $p(I_n|I_a)$ by PADS.* Left: log-scale over $p(I_n|I_a)$, right: original scale. Top row: learned sampling schedule (PADS); middle row: linear shift of a sampling interval from semihard[16] negatives to hard negatives; bottom row: shifting a static distance-based sampling[22] to gradually sample harder negatives.

Reward metrics e / Composition of state s	NMI	R@1	R@1 + NMI
Recall, Dist., NMI	63.9 68.5	65.5 68.9	**65.6** 69.2
Recall, Dist.	65.0 68.5	**65.7** **69.2**	64.4 **69.4**
Recall, NMI	63.7 68.4	63.9 68.2	64.2 68.5
Dist., NMI	65.3 68.8	65.3 68.7	65.1 68.5
Dist.	65.3 68.8	65.5 69.1	64.3 68.6
Recall	64.2 67.8	65.1 69.0	64.9 68.4
NMI	64.3 68.7	64.8 **69.2**	63.9 68.4

Table 3: Comparison of different compositions of the training state s and reward metric e. *Dist.* denotes average intra- and inter-class distances. Recall in state composition denotes all Recall@k-values, whereas for the target metric only Recall@1 was utilized.

M	10	30	50	70	100	∞	[22]
R@1	64.4	**65.7**	65.4	65.2	65.1	61.9	63.5
NMI	68.3	**69.2**	69.2	68.9	69.0	67.0	68.1

(a) Evaluation of the policy update frequency M.

$\mathcal{R}$	2	2, 32	2, 8, 16, 32	2, 8, 16, 32, 64
R@1	64.5	65.4	**65.7**	65.6
NMI	68.6	69.1	69.2	**69.3**

(b) Evaluation of various sets $\mathcal{R}$ of moving average lengths.

Table 4: Ablation experiments: (a) evaluates the influence of the number of DML iterations M performed before updating the policy π_θ using a reward r and, thus, the update frequency of π_θ. (b) analyzes the benefit of long-term learning progress information added to training states s by means of using various moving average lengths $\mathcal{R}$.

3. Comparison of RL Algorithms

We evaluate the applicability of the following RL algorithms for optimizing our policy π_θ (Eq. 4 in the main paper):

Approach	R@1	NMI
Margin[22]	63.5	68.1
REINFORCE	64.2	68.5
REINFORCE, EMA	64.8	68.9
REINFORCE, A2C	65.0	69.0
PPO, EMA	65.4	69.0
PPO, A2C	**65.7**	**69.2**
Q-Learn	63.2	67.9
Q-Learn, PR/2-Step	64.9	68.5

Table 5: Comparison of different RL algorithms. For policy-based algorithms (REINFORCE, PPO) we either use Exponential Moving Average (EMA) as a variance-reducing baseline or employ Advantage Actor Critic (A2C). In addition, we also evaluate Q-Learning methods (vanilla and Rainbow Q-Learning). For the Rainbow setup we use Priority Replay and 2-Step value approximation. Margin loss[22] is used as a representative reference for static sampling strategies.

- REINFORCE algorithm[21] with and without Exponential Moving Average (EMA)
- Advantage Actor Critic (A2C)[18]
- Rainbow Q-Learning[3] without extensions (vanilla) and using Priority Replay and 2-Step updates
- Proximal Policy Optimization (PPO)[17] applied to REINFORCE with EMA and to A2C.

For a comparable evaluation setting we use the CUB200-2011[19] dataset without learning rate scheduling and fixed 150 epochs of training. Within this setup, the hyperparameters related to each method are optimized via cross-validation. Tab. 5 shows that all methods, except for vanilla Q-Learning, result in an adjustment policy π_θ for $p(I_n|I_a)$ which outperforms static sampling strategies. Moreover, policy-based methods in general perform better than Q-Learning based methods with PPO being the best performing algorithm. We attribute this to the reduced search space (Q-Learning methods need to evaluate in state-actions space, unlike policy-methods, which work directly over the action space), as well as not employing replay buffers, i.e. not acting off-policy, since state-action pairs of previous training iterations may no longer be representative for current training stages.

4. Qualitative UMAP Visualization

Figure 2 shows a UMAP[10] embedding of test image features for CUB200-2011[19] learned by our model using PADS. We can see clear groupings for birds of the same and similar classes. Clusterings based on similar background is primarily due to dataset bias, e.g. certain types of birds occur only in conjunction with specific backgrounds.

5. Pseudo-Code

Algorithm 1 gives an overview of our proposed *PADS* approach using PPO with A2C as underlying RL method. Before training, our sampling distributions $p(I_n|I_a)$ is initialized with an initial distribution. Further, we initialize both the adjustment policy π_θ and the pre-update auxiliary policy π_θ^{old} for estimating the PPO probability ratio. Then, DML training is performed using triplets with random anchor-positive pairs and sampled negatives from the current sampling distribution $p(I_n|I_a)$. After M iterations, all reward and state metrics $\mathcal{E}, \mathcal{E}^*$ are computed on the embeddings $\phi(\cdot;\zeta)$ of $\mathcal{I}_{val}$. These values are aggregated in a training reward r and input state s. While r is used to update the current policy π_θ, s is fed into the updated policy to estimate adjustments a to the sampling distribution $p(I_n|I_a)$. Finally, after M^{old} iterations (e.g. we set to $M^{\text{old}} = 3$) π_θ^{old} is updated with the current policy weights θ.

6. Typical image retrieval failure cases

Fig. 3 shows nearest neighbours for good/bad test set retrievals. Even though the nearest neighbors do not always share the same class label as the anchor, all neighbors are very similar to the bird species depicted in the anchor images. Failures are due to very subtle differences.

Algorithm 1: Training one epoch via *PADS* by PPO

Input : $\mathcal{I}_{\text{train}}$, $\mathcal{I}_{\text{val}}$, Train labels $\mathcal{Y}_{\text{train}}$, Val. labels $\mathcal{Y}_{\text{val}}$, total iterations n_e

Parameter : Reward metrics $\mathcal{E}$, State metrics $\mathcal{E}^*$ + running average lengths $\mathcal{R}$, Num. of bins K, multiplier $\{\alpha, \beta\}$, $p_{\text{init}}(I_n|I_a)$, num. of iterations before updates M, M^{old}

```
// Initialization
p(I_n|I_a) ← p_init(I_n|I_a)
π_θ ← InitPolicy(K, α, β)
π_θ^old ← Copy(π_θ)

for i in n_e/M do
    // Update DML Model
    for j in M do
        // within batch B ∈ I_train
        I_a, I_p ← B
        I_n ~ p(I_n|I_a)
        ζ ← TrainDML({I_a, I_p, I_n}, φ(.; ζ))
    end

    // Update policy π_θ
    E_i ← E(I_val, Y_val, φ(.; ζ))
    E*_i ← E*(I_val, Y_val, φ(.; ζ))
    s_i ← GetState(E*_i, R, p(I_n|I_a))
    r ← GetReward(E_i, E_{i-1})
    l_π ← PPOLoss(π_θ, π_θ^old, s_{i-1}, a_{i-1})
    θ ← UpdatePolicy(l_π, π_θ)
    a_i ~ π_θ(a_i|s_i)
    p(I_n|I_a) ← Adjust(p(I_n|I_a), a_i)

    if i mod M^old == 0 then
        π_θ^old ← Copy(π)
    end
end
```

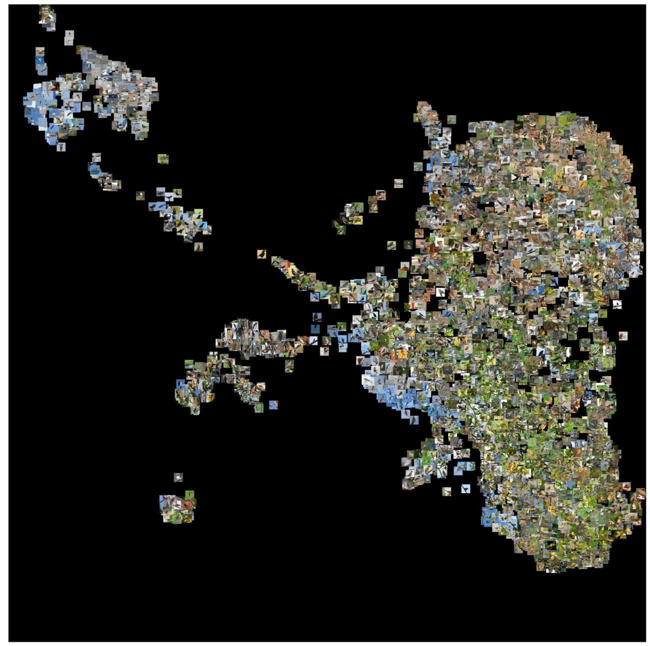

Figure 2: *UMAP embedding* based on the image embeddings $\phi(\cdot;\zeta)$ obtained from our proposed approach on CUB200-2011[19] (Test Set).

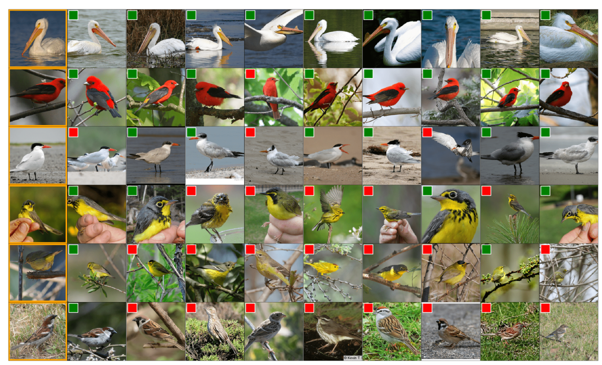

Figure 3: *Selection of good and bad nearest neighbour retrieval cases on CUB200-2011 (Test).* Orange bounding box marks query images, green/red boxes denote correct/incorrect retrievals.

References

[1] Weifeng Ge. Deep metric learning with hierarchical triplet loss. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 269–285, 2018. 2

[2] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006. 2

[3] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning, 2017. 3

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International Conference on Machine Learning*, 2015. 1

[5] Pierre Jacob, David Picard, Aymeric Histace, and Edouard Klein. Metric learning with horde: High-order regularizer

for deep embeddings. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[6] Wonsik Kim, Bhavya Goyal, Kunal Chawla, Jungmin Lee, and Keunjoo Kwon. Attention-based ensemble for deep metric learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015. 1

[8] 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. 1, 2

[9] Xudong Lin, Yueqi Duan, Qiyuan Dong, Jiwen Lu, and Jie Zhou. Deep variational metric learning. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2

[10] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861, 2018. 3

[11] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017. 1, 2

[12] Michael Opitz, Georg Waltner, Horst Possegger, and Horst Bischof. Deep metric learning with bier: Boosting independent embeddings robustly. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 2

[13] Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. 2019. 1, 2

[14] Karsten Roth, Biagio Brattoli, and Bjorn Ommer. Mic: Mining interclass characteristics for improved metric learning. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2

[15] Artsiom Sanakoyeu, Vadim Tschernezki, Uta Buchler, and Bjorn Ommer. Divide and conquer the embedding space for metric learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[16] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015. 2

[17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, 2017. 3

[18] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998. 3

[19] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 1, 2, 3, 4

[20] Xinshao Wang, Yang Hua, Elyor Kodirov, Guosheng Hu, Romain Garnier, and Neil M. Robertson. Ranked list loss for deep metric learning. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[21] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 1992. 3

[22] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017. 1, 2, 3

[23] Hong Xuan, Richard Souvenir, and Robert Pless. Deep randomized ensembles for metric learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 723–734, 2018. 2

[24] Yiru Zhao, Zhongming Jin, Guo-jun Qi, Hongtao Lu, and Xian-sheng Hua. An adversarial approach to hard triplet generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 501–517, 2018. 2

[25] Wenzhao Zheng, Zhaodong Chen, Jiwen Lu, and Jie Zhou. Hardness-aware deep metric learning. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2