A Additional Field Overview

A.1 Efficient Learning

Augmentations. Augmentation techniques are crucial in advancing deep learning models [50], providing strategies that enhance training efficiency, improve generalisation, and bolster model robustness. One of the recent groundworks includes batch augmentation [22], which is a powerful strategy that utilises large batches comprising multiple transformations for each sample. This not only accelerates training by reducing the number of stochastic gradient descent (SGD) updates but also acts as a regulariser, leading to improved generalisation. Another popular approach is RandAugment [10], designed for automated data augmentation by narrowing the search space and providing parameterisation. This method consistently outperforms previous automated augmentation techniques, demonstrating its efficiency across various tasks and datasets. Recent advances in contrastive learning also demonstrate that the so-called Stronger Augmentations [49] significantly enhance contrastive learning by enforcing distributional divergence between images augmented with such "strong" permutations as random cropping and flipping. Addressing limitations associated with regional dropout strategies, CutMix [54] involves cutting and pasting patches among training images. This ensures information preservation, promotes object localisation capabilities and improves the model's resilience against input corruption.

B Experimental Details

B.1 Datasets

Table 3: The details of three fine-grained visual classification datasets used for the experiments.

Dataset	Categories	Classes	Images
CUB-200-2011 [47]	Birds	200	11,788
Stanford Cars [28]	Cars	196	16,185
FGVC-Aircraft [33]	Airplanes	102	10,200

B.2 Implementation Details

For our self-distillation part, after performing random sampling from the input image, we resize both target and source sampled regions to 224×224 pixels. The motivation is to provide more different scales for input images so that the model can learn more scale-invariant representations, which are usually assumed to be already present in standard-sized datasets. Our training setup includes the standard SGD optimiser with a momentum equal to 0.9, a learning rate of 0.03, and a training batch size of 24 for all datasets. All experiments have been conducted on a single NVIDIA RTX 6000 GPU using the PyTorch framework and the APEX utility for mixed precision training.

C Additional Analysis

C.1 Qualitative Analysis

In order to analyse the motivation behind the significant performance improvement with our approach, we provide a direct comparison of training behaviour for a fine-tuned vanilla ResNet-50 and our self-distilled AD-Net with the same backbone. It can be observed from Figure 4 (left) that the model trained with standard fine-tuning procedure tends to overfit the limited data samples quickly and does not allow further refinement, while our framework is able to avoid overfitting more effectively. This can be explained by the effect of the additional distillation objective, which introduces an independent and more detailed source of information by enforcing feature space alignment for augmented views of the same image.

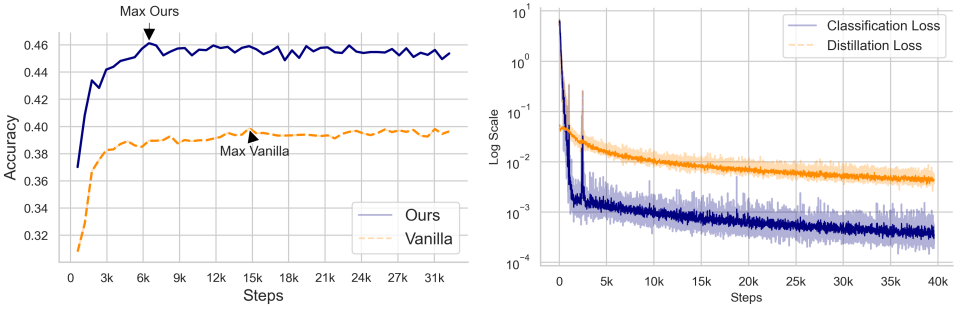

Figure 4: *Left*: Comparison of test accuracy evolution for our approach and traditionally fine-tuned vanilla ResNet-50 (CUB 10 % dataset). *Right*: Training loss evolution for both classification and distillation objectives in our AD-Net (ResNet-50, CUB 10 % dataset).

This specific effect is demonstrated in Figure 4 (right), where we provide the evolution of both classification and distillation objectives separately. Specifically, categorisation loss gets saturated quickly and becomes insignificant after the first 10% of training time, while our self-distillation component provides a noticeable effect throughout the whole training process. This is especially useful in low-data regimes, where models tend to quickly overfit to the main classification loss and do not obtain significant update signals, which is not the case for our proposed multi-component objective function.

C.2 Ablation Study

Distillation Loss. To rigorously evaluate the impact of distillation loss within our framework, a series of experiments were conducted on the CUB 10% dataset. First, in order to find the most effective type of distillation loss, we explore various objective functions applied at different type of outputs. In Table 4 we illustrate the variance in performance among different loss functions when applied to measure the disparity between features or logits coming from the target and source distillation branches in our architecture. Notably, Cross Entropy and Focal Loss functions, when applied to the output logits of both branches, demonstrate inferior performance. While computing the loss on the output features (converted to normalised distributions with softmax) is more effective, with KL divergence showing best results compared to the L1 and L2 objectives. One plausible explanation for the superior efficacy of KL divergence over other loss functions could be its ability to compare more sophisticated abstractions coming from differently augmented views.

Table 4: Effects caused by the type of distillation loss on the final metric. KL divergence computed between feature outputs of distillation branches shows the best performance.

Loss	Features	Logits
Cross Entropy	-	39.09
Focal Loss	-	39.32
L1 (MAE)	43.96	-
L2 (MSE)	41.67	-
Kullback–Leibler (KL)	**47.51**	-

Following the identification of the most effective loss function for our distillation branch, we proceeded to investigate its influence on the aggregated loss, denoted as $\mathcal{L}_{agg}$ in Eq. 3, by varying the weight coefficient α. In order to find the most optimal value, we experiment with both constant and variable α values, and summarise the results in Table 5. Our heuristic findings revealed that the model achieves its peak performance with the value of α set to 0.1.

Table 5: Study of the effect of α coefficient in the aggregated loss $\mathcal{L}_{agg}$ from Eq. 3. The α decay is a linear decay from 1.0 to 0.01 .

α	1	0.5	0.1	0.01	α decay
Acc, %	45.98	46.13	**47.51**	44.08	45.6

C.3 Limitations

Although our approach demonstrates a significant performance gain in the low-data setting, we also analyse and acknowledge its current limitations. First, due to the extra forward passes for each of the separate branches, the training time is increased by approximately 35-90 % compared to the vanilla fine-tuning (depending on the architecture type, see Tab. 2). However, this matter is not presented at inference time since our solution brings zero compute and time cost after training. Second, our approach has an inversely proportional performance gain to the size and diversity of a dataset (refer to Table 1), which theoretically may lead to less significant accuracy improvement when the data is abundant. Lastly, our solution requires a hyper-parameter α in Eq. 3 for controlling the influence of the distillation objective function on the overall loss, which unadapted value may sometimes lead to unstable training results (due to the nature of the KL divergence loss). We suggest that our current heuristic choice can be potentially replaced by an independent learnable parameter.

C.4 Quantitative Analysis

C.4.1 State-of-the-art comparison

Main Results. In Table 6 we provide the full comparison of different approaches on all data percentages including full datasets.

Other Results. Additionally, in Table 7, we also compare other methods potentially suitable for the low-data setting. Namely, SwAV [3], pre-trained in a self-supervised way and fine-tuned, and CLIP [36], a self-supervised vision-language model, used for zero-shot inference. As can be seen, compared to our AD-Net, they demonstrate promising but unstable results.

Table 6: Comparison of different approaches using various percentages of the data on three popular FGIC datasets. Our proposed solution achieves consistent improvement in performance over other methods across low data settings. Best results are highlighted in bold.

Dataset	Method	Training data percentage				
		10%	15%	30%	50%	100%
CUB-200-2011	ResNet-50	36.99	48.88	62.60	73.23	81.34
	FBP	37.88	49.12	63.27	73.70	82.52
	CBP-TS	37.12	47.82	62.24	72.37	81.48
	HBP	38.57	50.12	63.86	74.18	**86.12**
	DBTNet-50	37.67	49.52	63.16	73.28	86.04
	SAM (ResNet-50)	40.24	52.05	64.07	73.92	81.62
	SAM (FBP)	41.83	52.35	65.19	74.54	81.86
	Ours (ResNet-50)	**47.51**	**60.08**	**71.11**	**77.67**	82.06
Stanford Cars	ResNet-50	37.45	53.01	75.26	83.56	91.02
	FBP	40.13	55.07	76.42	85.10	91.63
	CBP-TS	37.77	54.87	75.51	84.80	89.52
	HBP	40.02	55.82	76.81	85.31	92.73
	DBTNet-50	39.48	55.24	76.52	86.52	**94.32**
	SAM (ResNet-50)	39.96	55.02	76.69	84.85	91.06
	SAM (FBP)	43.19	57.42	77.63	85.71	91.48
	Ours (ResNet-50)	**55.09**	**67.42**	**81.53**	**87.41**	91.96
FGVC-Aircraft	ResNet-50	43.52	53.17	71.32	78.61	87.13
	FBP	45.16	55.06	72.12	79.93	87.32
	CBP-TS	44.63	54.79	71.32	79.60	84.58
	HBP	45.28	56.12	72.58	81.47	89.74
	DBTNet-50	45.35	56.36	73.06	81.26	**90.86**
	SAM (ResNet-50)	46.73	56.02	72.59	79.21	86.74
	SAM (FBP)	47.97	57.47	73.43	80.86	87.46
	Ours (ResNet-50)	**55.81**	**62.59**	**74.44**	**81.73**	88.64

Table 7: Experiments with other models on datasets with 10% of training data. Results for CLIP are obtained using zero-shot classification. Vanilla and Our results are for comparison.

Type	Method	CUB	Cars	Air
CNN	ResNet-50 (vanilla) [21]	36.99	37.45	43.52
	SwAV (ResNet-50) [3]	16.91	36.19	49.49
	CLIP (ResNet-50, zero-shot) [36]	-	55.80	19.30
	Ours (ResNet-50)	47.51	55.09	55.81
ViT	ViT-B/32 (vanilla) [15]	65.60	28.21	33.84
	TransFG (ViT-B/32) [20]	64.91	-	-
	CLIP (ViT-B/32, zero-shot) [36]	-	59.40	21.20
	Ours (ViT-B/32)	69.27	33.34	36.01

C.4.2 Transferability

Additionally, in Table 8 we demonstrate the high transferability of our approach by utilising it on top of most of the popular CNN- and ViT-based backbones. The absolute improvement varies between 3-10 % showcasing that distilling local augmentations of input images indeed promotes feature refinement and is practically an architecture-independed technique.

Table 8: Transferability study of AD-Net on the CUB dataset with 10% of training data. Column Δ shows the absolute performance increase with our approach compared to a vanilla backbone.

Type	Backbone	Vanilla	Ours	Δ
CNN	ResNet-18	34.79	41.39	+6.60
	ResNet-34	36.83	45.81	+8.98
	ResNet-50	36.99	47.51	+10.52
	ResNet-101	40.19	49.44	+9.25
	GoogleNet	33.32	39.11	+5.79
	Inception v3	40.16	44.88	+4.72
	DenseNet-169	41.42	49.13	+7.71
ViT	ViT-B/32	65.60	69.27	+3.67
	FFVT B/32	65.79	68.13	+2.34

C.5 Augmentations with Naive Fine-tuning

Additionally, we conduct experiments with some advanced augmentation techniques, such as as ScaleMix [50], MultiCrop [2], and AsymAug [50] applied in a naive way with standard fine-tuning. As can be observed in Table 9, without our distillation technique, the ResNet-50 performance with the advanced augmentations is below the baseline with basic augmentations. Where by the basic augmentations we assume the classical list of augmentations recommended in the literature for each dataset. This includes random cropping, colour jittering, random horizontal flip, and further normalisation.

Table 9: The results with existing augmentation techniques, such as ScaleMix, MultiCrop, and AsymAug applied in a naive way with standard fine-tuning. Advanced augmentations include more geometrical and colour-related perturbations from the popular AutoAugs approach. As can be observed, without our distillation technique, the performance is below the baseline ResNet-50 with basic augmentations. The results were obtained with 10% low-data regime on the CUB training set.

Augmentation	Standard procedure	Our procedure
Basic augmentations	36.99	40.05
Advanced augmentations	27.82	30.68
ScaleMix [50]	28.54	26.99
MultiCrop [2]	28.06	28.90
AsymAugs [50]	30.29	30.96

We assume that more advanced and complex types of augmentations harm the learning process under low-data regimes, since the model may be unable to catch the locally important patterns due to harsh image perturbations.

C.6 Activation Maps Visualisation

In order to investigate the reason behind the significant performance improvement with our approach on the lowest data settings, we provide the difference in feature activation maps between the vanilla ResNet-50 and our AD-Net based on the same backbone. In Figures 5 and 6 we can clearly observe higher quality of the activation area from our method (more attention to the distinctive foreground regions), which explains its noticeable performance gain.

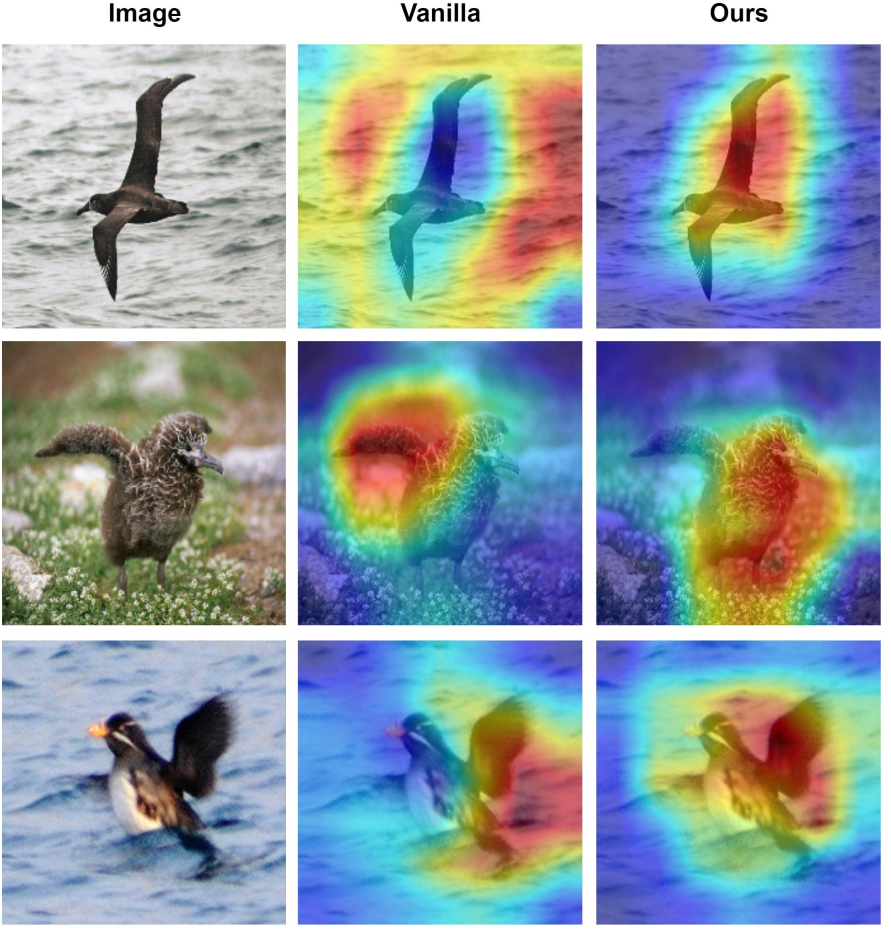

Figure 5: The visualisation of difference in feature activation maps between the vanilla ResNet-50 and our AD-Net on the CUB dataset. Red colour - higher activation, blue - lower activation.

Figure 6: The visualisation of difference in feature activation maps between the vanilla ResNet-50 and our AD-Net on the Stanford Cars dataset. Red colour - higher activation, blue - lower activation.

D Application Guidelines

D.1 Answers for Potential Questions

1. "How to decide whether should AD-Net be used for a given scenario?

A: Our method is supposed to be used in the scenarious where the standard fine-tuning procedure shows poor performance due to a small amount of available labelled images.

2. “How many images should be collected at least?”

A: After a thorough investigation, we have concluded that the exact answer depends on multiple factors, such as the number of classes, the number of images per class, and the size and capacity of a chosen baseline.

Therefore, we believe some small prior experiments are needed to make the final decision. Specifically, we recommend starting with at least 5 images per class, and further increasing the amount until the minimum desired performance is achieved.

Although the general rule is “the more data - the better”, our solution was designed specifically for the cases where obtaining a lot of labelled data may be impractical.

3. “When should AD-Net be switched to a different method as more training data are obtained?”

A: We suggest tracking the overall performance increase compared to an initially chosen baseline along with the data increase. Once the performance gain reaches neglectable values a different method can be used instead.

However, our solution is targeting the cases where the total amount of labelled data is highly limited and obtaining more samples may be too difficult.