A Appendix

In Appendix A.1 and Appendix A.2, we provide all the numbers of the figures in Section 4.1 and Section 4.3 separately. We show that SSL can also improve traditional fine-grained classification in Appendix A.3 and its model visualization in Appendix A.4. Last, we describe the implementation details in Appendix A.5.

A.1 Results on Few-shot Learning

Loss	*mini*-ImageNet	*tiered*-ImageNet	Birds	Cars	Aircrafts	Dogs	Flowers
	5-way 5-shot						
ProtoNet(PN)	75.2±0.6	75.9±0.7	87.3±0.5	91.7±0.4	91.4±0.4	83.0±0.6	89.2±0.6
PN+Jigsaw	76.2±0.6	78.0±0.7	89.8±0.4	92.4±0.4	91.8±0.4	85.7±0.5	**92.2±0.4**
Rel. err. red.	*4.0%*	*8.7%*	*19.7%*	*8.4%*	*4.7%*	*15.9%*	*27.8%*
Jigsaw	25.6±0.5	24.9±0.4	25.7±0.5	25.3±0.5	38.8±0.6	24.3±0.5	50.5±0.7
PN+Rotation	76.0±0.6	**78.9±0.7**	89.4±0.4	92.3±0.4	91.4±0.4	84.3±0.5	89.0±0.5
Rotation	51.4±0.7	50.7±0.8	33.1±0.6	29.4±0.5	29.5±0.5	27.3±0.5	49.4±0.7
PN+Jig.+Rot.	**76.6±0.7**	77.2±0.7	**90.2±0.4**	**92.7±0.4**	**91.9±0.4**	**85.9±0.5**	91.4±0.5
None	31.0±0.5	28.9±0.5	26.7±0.5	25.2±0.5	28.1±0.5	25.3±0.5	42.3±0.8
	20-way 5-shot						
ProtoNet(PN)	46.6±0.3	49.7±0.4	69.3±0.3	78.7±0.3	78.6±0.3	61.6±0.3	75.4±0.3
PN+Jigsaw	47.8±0.3	**52.4±0.4**	73.7±0.3	79.1±0.3	**79.1±0.2**	65.4±0.3	**79.2±0.3**
Jigsaw	9.2±0.2	7.5±0.1	8.1±0.1	7.1±0.1	15.4±0.2	7.1±0.1	25.7±0.2
PN+Rotation	48.2±0.3	**52.4±0.4**	72.9±0.3	**80.0±0.3**	78.4±0.2	63.4±0.3	73.9±0.3
Rotation	27.4±0.2	25.7±0.3	12.9±0.2	9.3±0.2	9.8±0.2	8.8±0.1	26.3±0.2
PN+Jig.+Rot.	**49.0±0.3**	51.2±0.4	**75.0±0.3**	79.8±0.3	79.0±0.2	**66.2±0.3**	78.6±0.3
None	10.8±0.1	11.0±0.2	9.3±0.2	7.5±0.1	8.9±0.1	7.8±0.1	22.6±0.2

Table 4: **Performance on few-shot learning tasks.** The mean accuracy (%) and the 95% confidence interval of 600 randomly chosen test experiments are reported for various combinations of loss functions. The top part shows the accuracy on 5-way 5-shot classification tasks, while the bottom part shows the same on 20-way 5-shot. Adding self-supervised losses to the ProtoNet loss improves the performance on all seven datasets on 5-way classification results. On 20-way classification, the improvements are even larger. The last row indicates results with a randomly initialized network. The top part of this table corresponds to Figure 2 in Section 4.1.

Table 4 shows the performance of ProtoNet with different self-supervision on seven datasets. We also test the accuracy of the model on novel classes when trained *only*

Loss	Birds Greyscale	Cars Low-resolution	Aircrafts Low-resolution	Dogs Greyscale	Flowers Greyscale
	5-way 5-shot				
ProtoNet	82.2±0.6	84.8±0.5	85.0±0.5	80.7±0.6	86.1±0.6
ProtoNet + Jigsaw	85.4±0.6	87.0±0.5	87.1±0.5	83.6±0.5	87.6±0.5
	20-way 5-shot				
ProtoNet	60.8±0.4	64.7±0.3	64.1±0.3	57.4±0.3	69.7±0.3
ProtoNet + Jigsaw	65.7±0.3	68.6±0.3	68.3±0.3	61.2±0.3	71.6±0.3

Loss	20% Birds	20% Cars	20% Aircrafts	20% Dogs	20% Flowers
	5-way 5-shot				
ProtoNet	73.0±0.7	75.8±0.7	77.7±0.6	68.5±0.7	82.2±0.7
ProtoNet + Jigsaw	75.4±0.7	82.8±0.6	78.4±0.6	69.1±0.7	86.0±0.6
	20-way 5-shot				
ProtoNet	46.4±0.3	51.8±0.4	52.3±0.3	40.8±0.3	62.8±0.3
ProtoNet + Jigsaw	49.8±0.3	61.5±0.4	53.6±0.3	42.2±0.3	68.5±0.3

Table 5: **Performance on *harder* few-shot learning tasks.** Accuracies are reported on novel set for 5-way 5-shot and 20-way 5-shot classification with degraded inputs, and with a subset (20%) of the images in the base set. The loss of color or resolution, and the smaller training set size make the tasks more challenging as seen by the drop in the performance of the ProtoNet baseline. However the improvements of using the *jigsaw puzzle loss* are higher in comparison to the results presented in Table 4.

with self-supervision on the base set of images. Compared to the randomly initialized model ("None" rows), training the network to predict rotations gives around 2% to 21% improvements on all datasets, while solving jigsaw puzzles only improves on aircrafts and flowers. However, these numbers are significantly worse than learning with supervised labels on the base set, in line with the current literature.

Table 5 shows the performance of ProtoNet with *jigsaw puzzle loss* on harder benchmarks. The results on the degraded version of the datasets are shown in the top part, and the bottom part shows the results of using only 20% of the images in the base categories. The gains using SSL are higher in this setting.

A.2 Results on Selecting Images for SSL

Table 6 shows the performance of selecting images for self-supervision, a tabular version of Figure 5 in Section 4.3. "Pool (random)" uniformly samples images proportional to the size of each dataset, while the "pool (weight)" one tends to pick more images from related domains.

Method	20% Birds	20% Cars	20% Aircrafts	20% Dogs	20% Flowers	20% *mini*-ImageNet
No SSL	73.0±0.7	75.8±0.7	77.7±0.6	68.5±0.7	82.2±0.7	67.81±0.65
SSL 20% dataset	74.4±0.7	82.1±0.6	77.7±0.6	71.8±0.7	85.9±0.6	68.47±0.66
SSL Pool (random)	74.1±0.7	78.4±0.7	78.8±0.6	68.5±0.7	83.5±0.7	68.94±0.68
SSL Pool (weight)	**76.4±0.6**	**82.9±0.6**	**80.2±0.6**	**72.4±0.7**	**87.6±0.6**	**69.81±0.65**
SSL 100% (oracle)	*78.4±0.6*	*83.7±0.6*	*81.5±0.6*	*74.7±0.6*	*88.1±0.6*	*70.13±0.67*

Table 6: **Performance on selecting images for self-supervision.** Adding more unlabeled images selected randomly from a pool often hurts the performance. Selecting similar images by importance weights improves on all five datasets.

Loss	Birds	Cars	Aircrafts	Dogs	Flowers
Softmax	47.0	72.6	69.9	51.4	72.8
Softmax + Jigsaw	49.2	73.2	**70.8**	53.5	**76.4**
Softmax + Rotation	**51.1**	**75.7**	70.0	**54.4**	73.5

Table 7: **Performance on standard fine-grained classification tasks.** Per-image accuracy (%) on the test set are reported. Using self-supervision improves the accuracy of a ResNet-18 network trained *from scratch* over the baseline of supervised training with cross-entropy (softmax) loss on all five datasets.

A.3 Results on Standard Fine-grained Classification

Here we present results on standard fine-grained classification tasks. Different from few-shot transfer learning, all the classes are seen in the training set and the test set contains novel images from the same classes. We use the standard training and test splits provided in the datasets. We investigate if SSL can improve the training of deep networks (*e.g.* ResNet-18 network) when *trained from scratch* (*i.e.* with random initialization) using images and labels in the training set only. The accuracy of using various loss functions are shown in Table 7. Training with self-supervision improves performance across datasets. On birds, cars, and dogs, predicting rotation gives 4.1%, 3.1%, and 3.0% improvements, while on aircrafts and flowers, the *jigsaw puzzle loss* yields 0.9% and 3.6% improvements.

A.4 Visualization of Learned Models

To understand why the representation generalizes, we visualize what pixels contribute the most to the correct classification for various models. In particular, for each image and model, we compute the gradient of the logits (predictions before softmax) for the correct class with respect to the input image. The magnitude of the gradient at each pixel is a proxy for its importance and is visualized as "saliency maps". Figure 7 shows these maps for various images and models trained with and without self-supervision on the standard classification task. It appears that the self-supervised models tend to

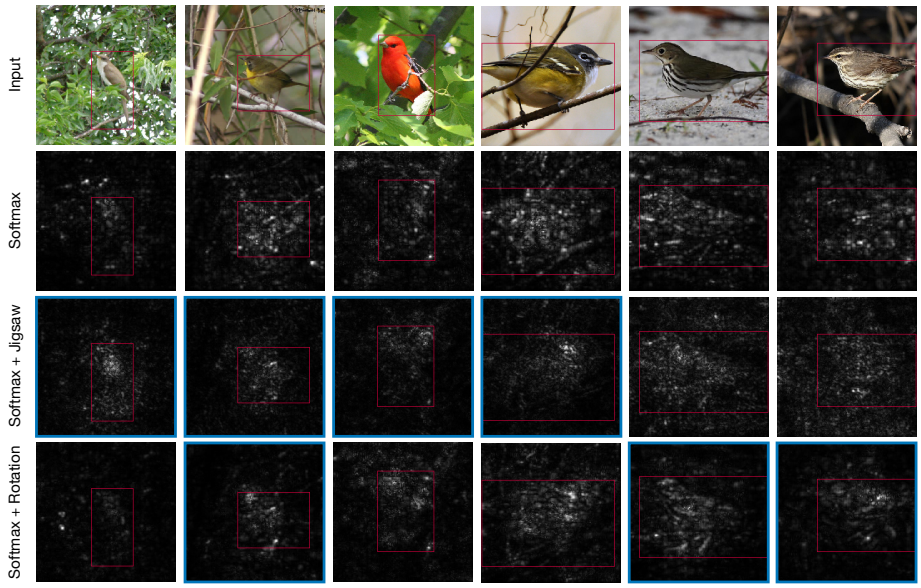

Fig. 7: **Saliency maps for various images and models.** For each image we visualize the magnitude of the gradient with respect to the correct class for models trained with various loss functions. The magnitudes are scaled to the same range for easier visualization. The models trained with self-supervision often have lower energy on the background regions when there is clutter. We highlight a few examples with blue borders and the bounding-box of the object for each image is shown in red.

focus more on the foreground regions, as seen by the amount of bright pixels within the bounding box. One hypothesis is that self-supervised tasks force the model to rely less on background features, which might be accidentally correlated to the class labels. For fine-grained recognition, localization indeed improves performance when training from few examples (see [64] for a contemporary evaluation of the role of localization for few-shot learning).

A.5 Experimental Details

Optimization details on few-shot learning During training, especially for the jigsaw puzzle task, we found it to be beneficial to *not* track the running mean and variance for the batch normalization layer, and instead estimate them for each batch independently. We hypothesize that this is because the inputs contain both full-sized images and small patches, which might have different statistics. At test time we do the same. We found the accuracy goes up as the batch size increases but saturates at a size of 64.

When training with supervised and self-supervised loss, a trade-off term λ between the losses can be used, thus the total loss is $\mathcal{L} = (1 - \lambda)\mathcal{L}_s + \lambda\mathcal{L}_{ss}$. We find that simply use $\lambda = 0.5$ works the best, except for training on *mini*- and *tiered*-ImageNet with

jigsaw loss, where we set $\lambda = 0.3$. We suspect that this is because the variation of the image size and the categories are higher, making the self-supervision harder to train with limited data. When both jigsaw and rotation losses are used, we set $\lambda = 0.5$ and the two self-supervised losses are averaged for $\mathcal{L}_{ss}$.

For training meta-learners, we use 16 query images per class for each training episode. When only 20% of labeled data are used, 5 query images per class are used. For MAML, we use 10 query images and the approximation method for backpropagation as proposed in [10] to reduce the GPU memory usage. When training with self-supervised loss, it is added when computing the loss in the outer loop. We use PyTorch [45] for our experiments.

Optimization details on domain classifier For the domain classifier, we first obtain features from the penultimate-layer (2048 dimensional) from a ResNet-101 model pre-trained on ImageNet [52]. We then train a binary logistic regression model with weight decay using LBFGS for 1000 iterations. The images from the labeled dataset are the positive class and from the pool of unlabeled data are the negative class. A subset of negative images are selected uniformly at random with 10 times the size of positive images. A loss for the positive class is scaled by the inverse of its frequency to account for the significantly larger number of negative examples.

Optimization details on standard classification For standard classification (Appendix A.3) we train a ResNet-18 network *from scratch*. All the models are trained with ADAM optimizer with a learning rate of 0.001 for 600 epochs with a batch size of 16. We track the running statistics for the batch normalization layer for the softmax baselines following the conventional setting, *i.e.* w/o self-supervised loss, but do not track these statistics when training with self-supervision.

Architectures for self-supervised tasks For jigsaw puzzle task, we follow the architecture of [41] where it was first proposed. The ResNet18 results in a 512-dimensional feature for each input, and we add a fully-connected (`fc`) layer with 512-units on top. The nine patches give nine 512-dimensional feature vectors, which are concatenated. This is followed by a `fc` layer, projecting the feature vector from 4608 to 4096 dimensions, and a `fc` layer with 35-dimensional outputs corresponding to the 35 permutations for the jigsaw task.

For rotation prediction task, the 512-dimensional output of ResNet-18 is passed through three `fc` layers with {512, 128, 128, 4} units. The predictions correspond to the four rotation angles. Between each `fc` layer, a `ReLU` activation and a dropout layer with a dropout probability of 0.5 are added.