A. Experimental Details

The mini-ImageNet and CIFAR-FS datasets can be found at https://github.com/yaoyao-liu/mini-imagenet-tools and https://github.com/ArnoutDevos/maml-cifar-fsrespectively.

A.1. Mixing Meta-Learned Models and Fine-Tuning Procedures: Additional Experiments

Model	SVM	RR	ProtoNet	MAML
MetaOptNet-M	**78.63** $\pm$ 0.25 %	**76.96** $\pm$ 0.23 %	**76.17** $\pm$ 0.23 %	70.14 $\pm$ 0.27 %
MetaOptNet-C	76.72 $\pm$ 0.24 %	74.48 $\pm$ 0.24 %	73.37 $\pm$ 0.24 %	**71.32** $\pm$ 0.26 %
R2-D2-M	**68.40** $\pm$ 0.20 %	**72.09** $\pm$ 0.25 %	**70.74** $\pm$ 0.25 %	**71.43** $\pm$ 0.27 %
R2-D2-C	68.24 $\pm$ 0.26 %	67.04 $\pm$ 0.26 %	60.93 $\pm$ 0.29 %	65.30 $\pm$ 0.27 %

Table 7. Comparison of meta-learning and transfer learning models with various fine-tuning algorithms on 5-shot mini-ImageNet. "MetaOptNet-M" and "MetaOptNet-C" denote models with MetaOptNet backbone trained with MetaOptNet-SVM and classical training. Similarly, "R2-D2-M" and "R2-D2-C" denote models with R2-D2 backbone trained with ridge regression (RR) and classical training. Column headers denote the fine-tuning algorithm used for evaluation, and the radius of confidence intervals is one standard error.

A.2. Transfer Learning and Feature Space Clustering

We evaluate the proposed regularizers and classically trained baseline on two backbone architectures: a 4-layer convolutional neural network with number of filters per layer 96-192-384-512 originally used for R2-D2 (Bertinetto et al., 2018) and ResNet-12 (He et al., 2016; Oreshkin et al., 2018; Lee et al., 2019). We run experiments on the mini-ImageNet and CIFAR-FS datasets.

When training the backbone feature extractors, we use SGD with a batch-size of 128 for CIFAR-FS and 256 for mini-ImageNet, Nesterov momentum set to 0.9 and weight decay of 10^{-4}. For training on CIFAR-FS, we set the initial learning rate to 0.1 for the first 100 epochs and reduce by a factor of 10 every 50 epochs. To avoid gradient explosion problems, we use 15 warm-up epochs for mini-ImageNet with learning rate 0.01. We train all classically trained networks for a total of 300 epochs. We employ data parallelism across 2 Nvidia RTX 2080 Ti GPUs when training on mini-ImageNet, and we only use one GPU for each CIFAR-FS experiment. For few-shot testing, we train two classification heads, a linear NN layer and SVM (Lee et al., 2019) on top of the pre-trained feature extractors. The evaluation results of these models are given in Table 9. Table 8 shows the running time per training epoch as well as total training time on both datasets and backbone architectures to achieve the results in Table 3. The training speed of the proposed regularizers is nearly as fast as classical transfer learning and up to almost 13 times faster than meta-learning methods. For meta-learning methods, we follow the training hyperparemeters from (Lee et al., 2019).

		mini-ImageNet	CIFAR-FS
Training	Backbone	runtime	runtime
R2-D2	R2-D2	16m/16.8h	44s/45m
Classical	R2-D2	20s/1.7h	4s/22m
Classical w/ R_{FC}	R2-D2	20s/1.7h	4s/24m
Classical w/ R_{HV}	R2-D2	20s/1.7h	4s/23m
MetaOptNet-SVM	MetaOptNet	1.5h/88.0h	4m/4.5h
Classical	MetaOptNet	1.4m/7.0h	14s/1.2h
Classical w/ R_{FC}	MetaOptNet	1.5m/7.4h	15s/1.3h
Classical w/ R_{HV}	MetaOptNet	1.3m/7.2h	16s/1.4h

Table 8. Runtime (training time per epoch/total times) comparison of methods on CIFAR-FS and mini-ImageNet 5-way classification on a single GPU.

				mini-ImageNet		CIFAR-FS	
Backbone	Regularizer	Coeff	Head	1-shot	5-shot	1-shot	5-shot
R2-D2	R_{FC}	0.02	NN	$48.27 \pm 0.29\%$	$69.13 \pm 0.26\%$	$63.11 \pm 0.35\%$	$83.31 \pm 0.25\%$
		0.05	NN	$48.75 \pm 0.29\%$	$69.50 \pm 0.26\%$	$64.49 \pm 0.35\%$	$83.32 \pm 0.25\%$
		0.1	NN	$48.72 \pm 0.29\%$	$67.39 \pm 0.25\%$	$62.98 \pm 0.36\%$	$81.07 \pm 0.26\%$
	R_{HV}	0.02	NN	$46.74 \pm 0.28\%$	$68.19 \pm 0.27\%$	$62.50 \pm 0.34\%$	$82.90 \pm 0.25\%$
		0.05	NN	$49.11 \pm 0.29\%$	$68.88 \pm 0.26\%$	$63.61 \pm 0.35\%$	83.21 ± 0.25 %
		0.1	NN	$48.87 \pm 0.29\%$	$69.67 \pm 0.26\%$	$63.50 \pm 0.35\%$	$83.17 \pm 0.25\%$
	R_{FC}	0.02	SVM	$49.05 \pm 0.30\%$	$68.94 \pm 0.26\%$	$64.48 \pm 0.34\%$	$83.11 \pm 0.25\%$
		0.05	SVM	$50.39 \pm 0.30\%$	$69.58 \pm 0.26\%$	$65.53 \pm 0.35\%$	$83.30 \pm 0.25\%$
		0.1	SVM	$50.71 \pm 0.30\%$	$68.46 \pm 0.25\%$	$64.25 \pm 0.36\%$	$81.57 \pm 0.26\%$
	R_{HV}	0.02	SVM	$47.81 \pm 0.29\%$	$68.08 \pm 0.27\%$	$63.71 \pm 0.33\%$	$82.77 \pm 0.26\%$
		0.05	SVM	$49.28 \pm 0.30\%$	$68.62 \pm 0.26\%$	$64.52 \pm 0.34\%$	$82.99 \pm 0.26\%$
		0.1	SVM	$50.16 \pm 0.30\%$	$69.54 \pm 0.26\%$	$64.62 \pm 0.34\%$	$83.08 \pm 0.26\%$
ResNet-12	R_{FC}	0.02	NN	$57.54 \pm 0.32\%$	$77.31 \pm 0.25\%$	$71.69 \pm 0.36\%$	$86.13 \pm 0.23\%$
		0.05	NN	$56.59 \pm 0.33\%$	$74.81 \pm 0.25\%$	$71.78 \pm 0.37\%$	$85.30 \pm 0.24\%$
		0.1	NN	$52.26 \pm 0.35\%$	$69.93 \pm 0.28\%$	$71.85 \pm 0.39\%$	$83.74 \pm 0.25\%$
	R_{HV}	0.02	NN	$53.75 \pm 0.30\%$	$76.11 \pm 0.25\%$	$70.12 \pm 0.35\%$	$86.37 \pm 0.23\%$
		0.05	NN	$57.15 \pm 0.31\%$	$77.27 \pm 0.25\%$	$71.49 \pm 0.36\%$	$85.85 \pm 0.24\%$
		0.1	NN	$57.76 \pm 0.33\%$	$76.05 \pm 0.26\%$	$71.56 \pm 0.37\%$	$84.80 \pm 0.25\%$
	R_{FC}	0.02	SVM	$59.38 \pm 0.31\%$	$78.15 \pm 0.24\%$	$72.32 \pm 0.30\%$	$86.31 \pm 0.24\%$
		0.05	SVM	$59.05 \pm 0.32\%$	$76.36 \pm 0.24\%$	$71.94 \pm 0.36\%$	$85.28 \pm 0.24\%$
		0.1	SVM	$56.73 \pm 0.35\%$	$73.70 \pm 0.26\%$	$71.08 \pm 0.36\%$	$83.49 \pm 0.25\%$
	R_{HV}	0.02	SVM	$56.95 \pm 0.30\%$	$77.06 \pm 0.24\%$	$71.34 \pm 0.35\%$	$86.54 \pm 0.23\%$
		0.05	SVM	$59.36 \pm 0.31\%$	$77.97 \pm 0.24\%$	$72.00 \pm 0.36\%$	$85.87 \pm 0.24\%$
		0.1	SVM	$59.37 \pm 0.32\%$	$77.05 \pm 0.25\%$	$71.92 \pm 0.37\%$	$84.84 \pm 0.25\%$

Table 9. Hyper-parameter tuning for R_{FC} and R_{HV} regularizers with various backbone structures and classification heads on 1-shot and 5-shot CIFAR-FS and mini-ImageNet 5-way classification. Regularizer coefficients include the C/N factor.

A.3. Reptile Weight Clustering

We train models via our weight-clustering Reptile algorithm with a range of coefficients for the regularization term. The model architecture and all other hyperparameters were chosen to match those specified for Reptile training and evaluation on 1-shot and 5-shot mini-ImageNet in (Nichol & Schulman, 2018). The evaluation results of these models are given in Table 10. All models were trained on Nvidia RTX 2080 Ti GPUs.

Coefficient	1-shot	5-shot
0 (Reptile)	$49.97 \pm 0.32\%$	$65.99 \pm 0.58\%$
1.0×10^{-5}	$51.42 \pm 0.23\%$	$67.16 \pm 0.22\%$
2.5×10^{-5}	$51.25 \pm 0.24\%$	$67.55 \pm 0.22\%$
5.0×10^{-5}	$\mathbf{51.94} \pm 0.23\%$	$\mathbf{68.02} \pm 0.22\%$
7.5×10^{-5}	$51.40 \pm 0.24\%$	$67.59 \pm 0.22\%$
1.0×10^{-4}	$50.92 \pm 0.23\%$	$67.91 \pm 0.22\%$
2.5×10^{-4}	$50.65 \pm 0.23\%$	$65.95 \pm 0.23\%$
5.0×10^{-4}	$51.37 \pm 0.23\%$	$66.98 \pm 0.23\%$

Table 10. Comparison of test accuracy for models trained with the weight-clustering Reptile algorithm with various regularization coefficients evaluated on 1-shot and 5-shot mini-ImageNet tasks. The results for vanilla Reptile are those given in (Nichol & Schulman, 2018).

A.4. Architectures

For our experiments using MAML, R2-D2, MetaOptNet, and Reptile, we use the architectures originally used for experiments in the respective papers (Finn et al., 2017; Bertinetto et al., 2018; Lee et al., 2019; Nichol & Schulman, 2018). Specificaly, (Finn et al., 2017; Nichol & Schulman, 2018) use the same network with 4 convolutional layers. (Bertinetto et al., 2018) uses a modified version of this convolutional network, while (Lee et al., 2019) employs a ResNet-12 architecture.

B. Proof of Theorem 1

Consider the three conditions

$$\|x - \overline{X}\| < \delta, \quad \|y - \overline{Y}\| < \delta, \quad \|z - \overline{X}\| < \delta,$$

where $\delta = \|\overline{X} - \overline{Y}\|/4$, and $\overline{X}$ is the expected value of X. Under these conditions,

$$\|z - x\| \leq \|z - \overline{X}\| + \|x - \overline{X}\| < 2\delta$$

and

$$\|z - y\| \geq \|\overline{X} - \overline{Y}\| - \|y - \overline{Y}\| - \|z - \overline{X}\| > 4\delta - 2\delta = 2\delta.$$

Combining the above yields

$$\|z - x\| < \|z - y\|.$$

We can now write

$$\begin{aligned} z^T(x-y) - \frac{1}{2}\|x\|^2 + \frac{1}{2}\|y\|^2 & \\ &= -\|z-x\|^2 + \frac{1}{2}\|z-y\|^2 + \frac{1}{2}\|z-x\|^2 \\ &\geq -\|z-x\|^2 + \frac{1}{2}\|z-x\|^2 + \frac{1}{2}\|z-x\|^2 \\ &= 0, \end{aligned}$$

and so z is classified correctly if our three conditions hold. From the Chebyshev bound, these conditions hold with probability at least

$$\left(1 - \frac{\sigma_x^2}{\delta^2}\right)^2 \left(1 - \frac{\sigma_y^2}{\delta^2}\right) \geq \left(1 - \frac{2\sigma_x^2}{\delta^2}\right)\left(1 - \frac{\sigma_y^2}{\delta^2}\right) \geq 1 - \frac{2\sigma_x^2 + \sigma_y^2}{\delta^2}, \tag{1}$$

where we have twice applied the identity $(1-a)(1-b) \geq (1-a-b)$, which holds for $a, b \geq 0$, (this also requires $\sigma_y^2/\delta^2 < 1$, but this can be guaranteed by choosing a sufficiently small ϵ as in the statement of the theorem).

Finally, we have the variation ratio bound

$$\frac{\text{var}[X] + \text{var}[Y]}{\text{var}[U]} = \frac{\sigma_x^2 + \sigma_y^2}{\sigma_x^2 + \sigma_y^2 + 16\delta^2} < \epsilon.$$

And so

$$\delta^2 \geq \frac{(1-\epsilon)(\sigma_x^2 + \sigma_y^2)}{16\epsilon}.$$

Plugging this into (1) we get the final probability bound

$$1 - \frac{32\epsilon\sigma_x^2 + 16\epsilon\sigma_y^2}{(\sigma_x^2 + \sigma_y^2)(1-\epsilon)} \geq 1 - \frac{32\epsilon\sigma_x^2 + 32\epsilon\sigma_y^2}{(\sigma_x^2 + \sigma_y^2)(1-\epsilon)} = 1 - \frac{32\epsilon}{(1-\epsilon)}. \tag{2}$$