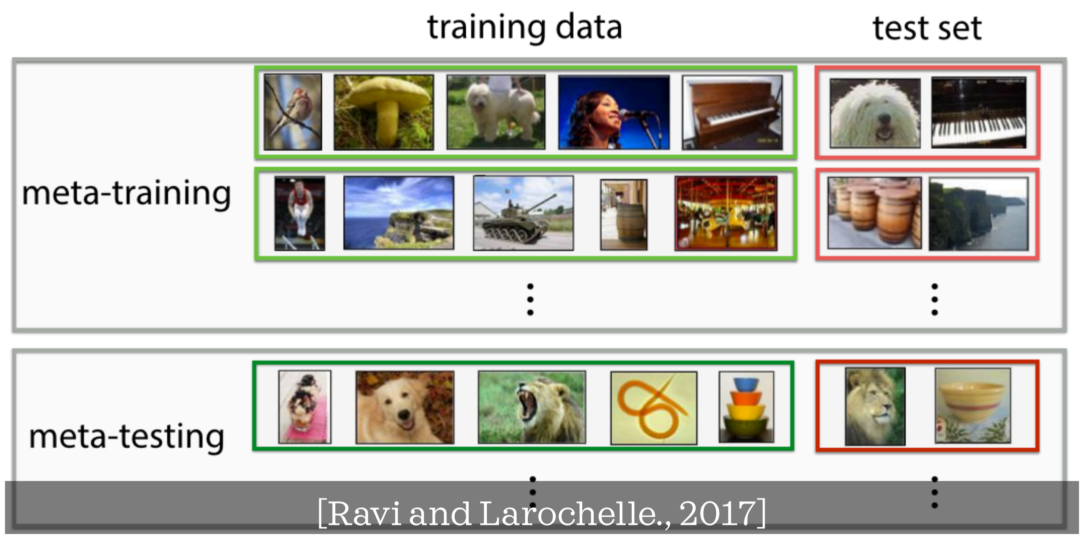
Unraveling meta-learning

Understanding feature representation for few-shot tasks

Goldblum et al., ICML 2020



[Ravi and Larochelle., 2017]

Introduction

Meta-learning:

learning to learn (or optimize)

> learn a function f that is a set of learning algorithm F or feature extractor

N-ways *K-shot* classification:

In each training and test tasks, there are **N classes**, each has *K examples*.

Research Question

Are Meta-Learned Features Fundamentally Better for Few-Shot Learning?

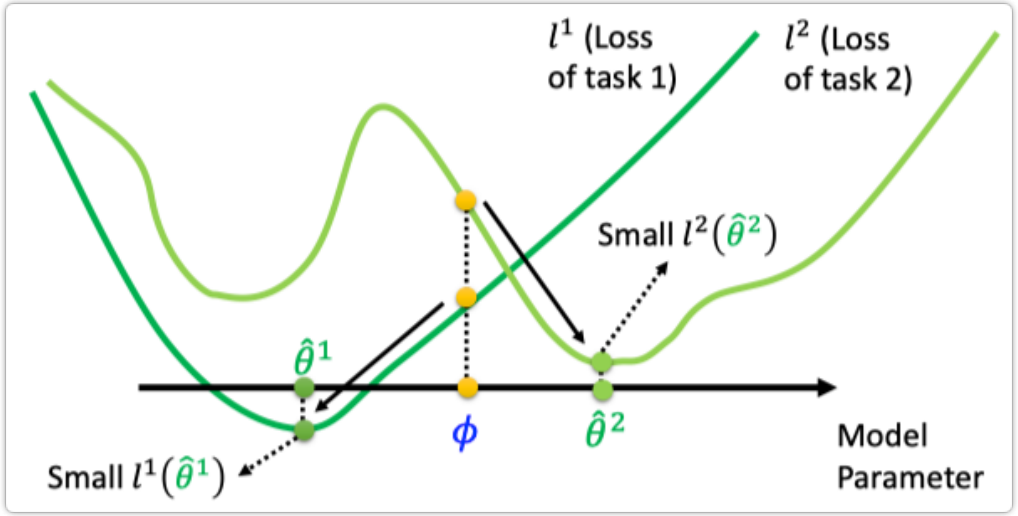
> the differences between features learned by meta-learning and classical training;

> explore the different methods with two proposed mechanisms (regularizers)

Introduction

Two categories:

- tune the feature extractor (e.g., MAML & Reptile)
  - search for meta-parameters that lie close in weight space to a wide range of task-specific minima
- fix the feature extractor (e.g., R2-D2 and MetaOptNet)
  - cluster object classes more tightly in feature space



Gradient-based optimization (Hong-yi Lee)

MAML vs. Reptile

Algorithm 2 Reptile, batched version

Initialize ϕ
for iteration $= 1, 2, \ldots$ **do**
  Sample tasks $\tau_1, \tau_2, \ldots, \tau_n$
  for $i = 1, 2, \ldots, n$ **do**
    Compute $W_i = \mathrm{SGD}(L_{\tau_i}, \phi, k)$
  end for
  Update $\phi \leftarrow \phi + \epsilon \frac{1}{k} \sum_{i=1}^{n} (W_i - \phi)$
end for

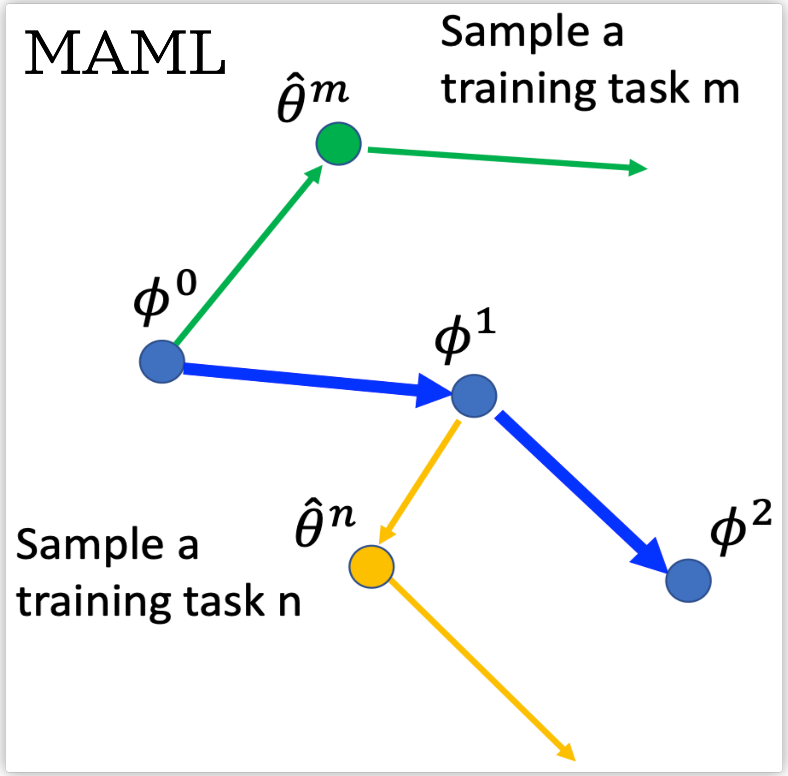
[Nichol & Schulman, 2018]

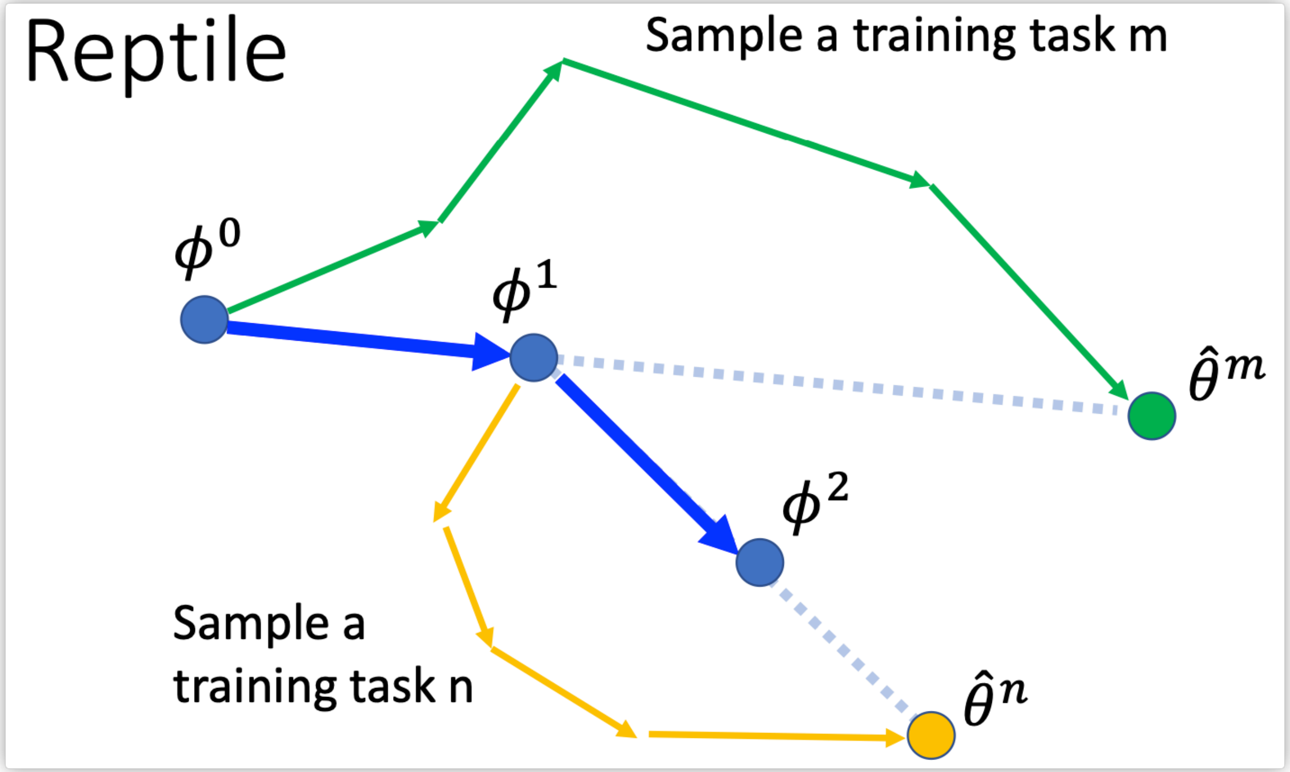
Algorithm 2 MAML for Few-Shot Supervised Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
randomly initialize θ
while not done **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\mathcal{T}_i$ **do**
    Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$
    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation (2) or (3)
    Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
    Sample datapoints $\mathcal{D}'_i = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$ for the meta-update
  end for
  Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})$ using each $\mathcal{D}'_i$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 2 or 3
end while

[Finn et al., 2017]

MAML vs. Reptile





[Hong-yi Lee]

Last-layer Methods

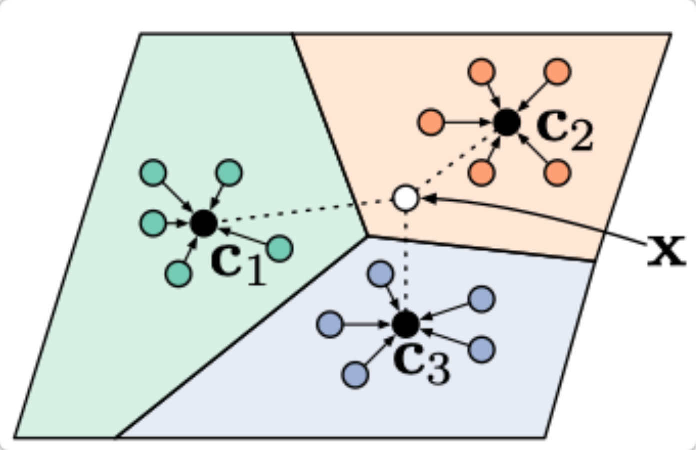
use differentiable optimizers to minimize the fine-tuning objective and then differentiate the solution with respect to feature inputs.

- R2-D2 (Bertinetto et al. 2018): Ridge Regression Differentiable Discriminator

$$\arg\min_{W} \|XW - Y\|^2 + \lambda \|W\|^2$$

- MetaOptNet (Lee et al., 2019): SVM

- ProtoNet (Snell et al., 2017): the proximity of input features to class centroids



Class Clustering in Feature Space

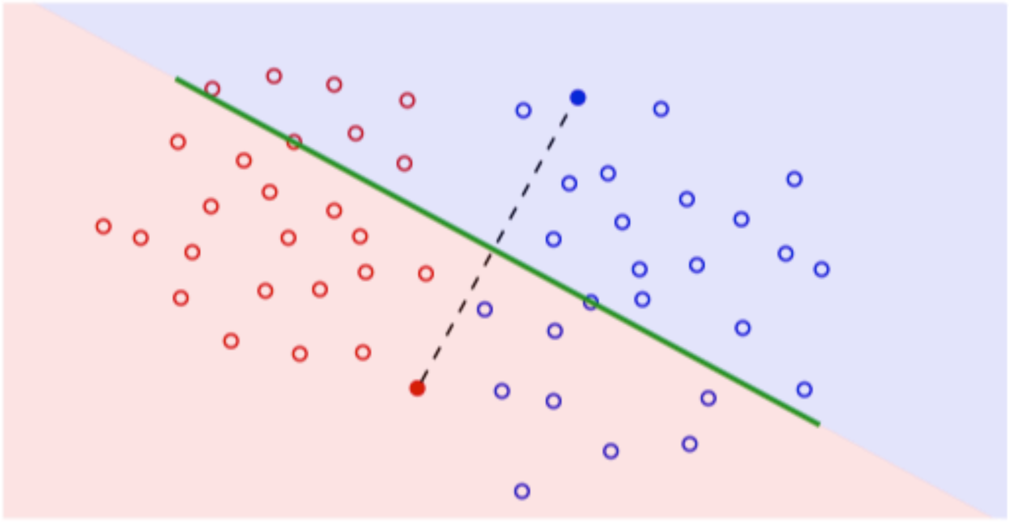
Conclusion: meta-learned models separate features differently than classically trained networks.

R_{FC}: measurement of feature clustering

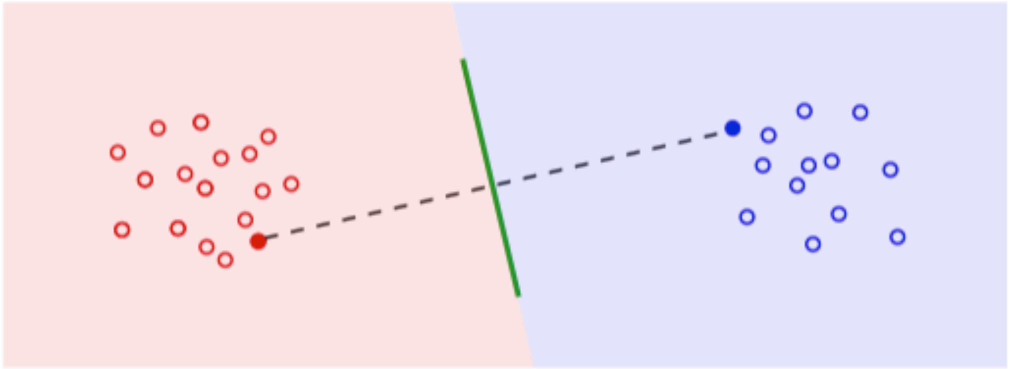
R_{HV}: measurement of hyperplane variation

Training	Dataset	R_{FC}	R_{HV}
R2-D2-M	CIFAR-FS	**1.29**	**0.95**
R2-D2-C	CIFAR-FS	2.92	1.69
MetaOptNet-M	CIFAR-FS	**0.99**	**0.75**
MetaOptNet-C	CIFAR-FS	1.84	1.25
R2-D2-M	mini-ImageNet	**2.60**	**1.57**
R2-D2-C	mini-ImageNet	3.58	1.90
MetaOptNet-M	mini-ImageNet	**1.29**	**0.95**
MetaOptNet-C	mini-ImageNet	3.13	1.75

Comparison of class separation metrics for feature extractors trained by classical and meta-learning routines.



(a)



(b)

Linear Separability

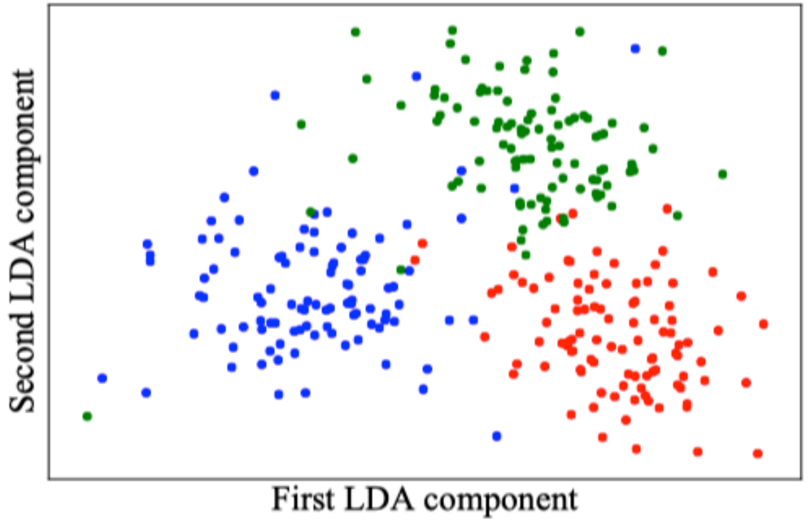
(a) class variation is high relative to the variation between classes

(b) classes move farther apart relative to the class variation

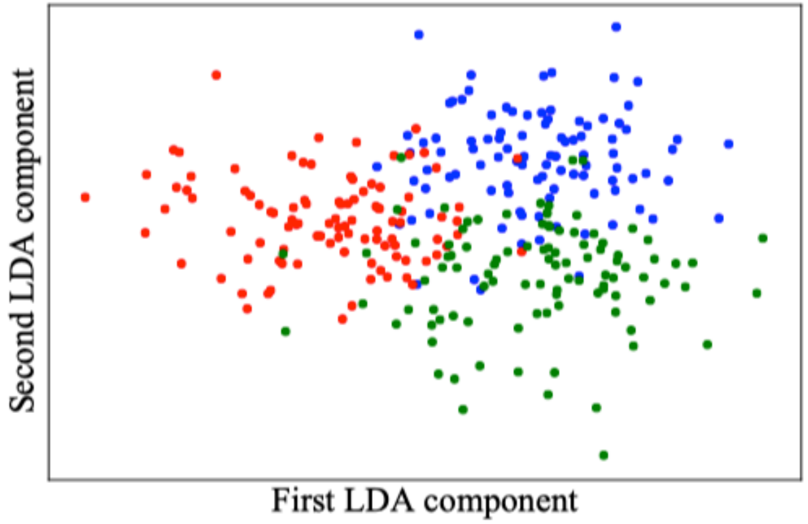
Meta v.s Classical

- classically trained model mashes features together

- the meta-learned models draws the classes farther apart



(a) Meta-Learning



(b) Classically Trained

Feature Clustering Regularizer

R_{FC}: the measurement of feature clustering

$$R_{FC}(\theta, \{x_{i,j}\}) = \frac{C}{N} \frac{\sum_{i,j} \|f_\theta(x_{i,j}) - \mu_i\|_2^2}{\sum_i \|\mu_i - \mu\|_2^2}$$

$f_\theta(x_{i,j})$: feat. vec. for data j in cls i;
u_i: mean of feat. vec. in class i;
u: mean across all feature vectors.

Feature space clustering improves few-shot performance of transfer learning

Hyperplane Variation Regularizer

R_{HV}: measurement of hyperplane variation

$$R_{HV}(f_\theta(x_1), f_\theta(x_2), f_\theta(y_1), f_\theta(y_2)) = \frac{\|(f_\theta(x_1) - f_\theta(y_1)) - (f_\theta(x_2) - f_\theta(y_2))\|_2}{\|(f_\theta(x_1) - f_\theta(y_1)\|_2 + \|f_\theta(x_2) - f_\theta(y_2)\|_2}$$

x1, x2 in class A;
y1, y2 in class B

Distance between distance vectors $x_1 - y_1$ and $x_2 - y_2$ relative to their size.

Experiements

Training	Backbone	mini-ImageNet		CIFAR-FS	
		1-shot	5-shot	1-shot	5-shot
R2-D2	R2-D2	$\mathbf{51.80} \pm 0.20\%$	$68.40 \pm 0.20\%$	$65.3 \pm 0.2\%$	$79.4 \pm 0.1\%$
Classical	R2-D2	$48.39 \pm 0.29\%$	$68.24 \pm 0.26\%$	$62.9 \pm 0.3\%$	$82.8 \pm 0.3\%$
Classical w/ R_{FC}	R2-D2	$50.39 \pm 0.30\%$	$\mathbf{69.58} \pm 0.26\%$	$\mathbf{65.5} \pm 0.4\%$	$\mathbf{83.3} \pm 0.3\%$
Classical w/ R_{HV}	R2-D2	$50.16 \pm 0.30\%$	$69.54 \pm 0.26\%$	$64.6 \pm 0.3\%$	$83.1 \pm 0.3\%$
MetaOptNet-SVM	MetaOptNet	$\mathbf{62.64} \pm 0.31\%$	$\mathbf{78.63} \pm 0.25\%$	$72.0 \pm 0.4\%$	$84.2 \pm 0.3\%$
Classical	MetaOptNet	$56.18 \pm 0.31\%$	$76.72 \pm 0.24\%$	$69.5 \pm 0.3\%$	$85.7 \pm 0.2\%$
Classical w/ R_{FC}	MetaOptNet	$59.38 \pm 0.31\%$	$78.15 \pm 0.24\%$	$\mathbf{72.3} \pm 0.4\%$	$\mathbf{86.3} \pm 0.2\%$
Classical w/ R_{HV}	MetaOptNet	$59.37 \pm 0.32\%$	$77.05 \pm 0.25\%$	$72.0 \pm 0.4\%$	$85.9 \pm 0.2\%$

Table 3. Comparison of methods on 1-shot and 5-shot CIFAR-FS and mini-ImageNet 5-way classification. The top accuracy for each backbone/task is in bold. Confidence intervals have radius equal to one standard error. Few-shot fine-tuning is performed with SVM except for R2-D2, for which we report numbers from the original paper.

Weight-Clustering

- ❖ Finding clusters of local minima for task losses in parameter space
- ❖ Reptile: minimizing the consensus formulation

$$\frac{1}{m}\sum_{p=1}^{m}\mathcal{L}_{\mathcal{T}_p}(\tilde{\theta}_p) + \frac{\gamma}{2}\|\tilde{\theta}_p - \theta\|^2$$

$$\theta \leftarrow \theta - \eta\tilde{\theta}_p$$

- ❖ Weight-Clustering Regularization

$$R_i\big(\{\tilde{\theta}_p\}_{p=1}^{m}\big) = d\big(\tilde{\theta}_i, \frac{1}{m}\sum_{p=1}^{m}\tilde{\theta}_p\big)^2$$

- ❖ Inner-loop optimization

$$\mathcal{L} = \mathcal{L}_{\mathcal{T}_i}^{j} + \alpha R_i\big(\{\tilde{\theta}_p^{j-1}\}_{p=1}^{m}\big)$$

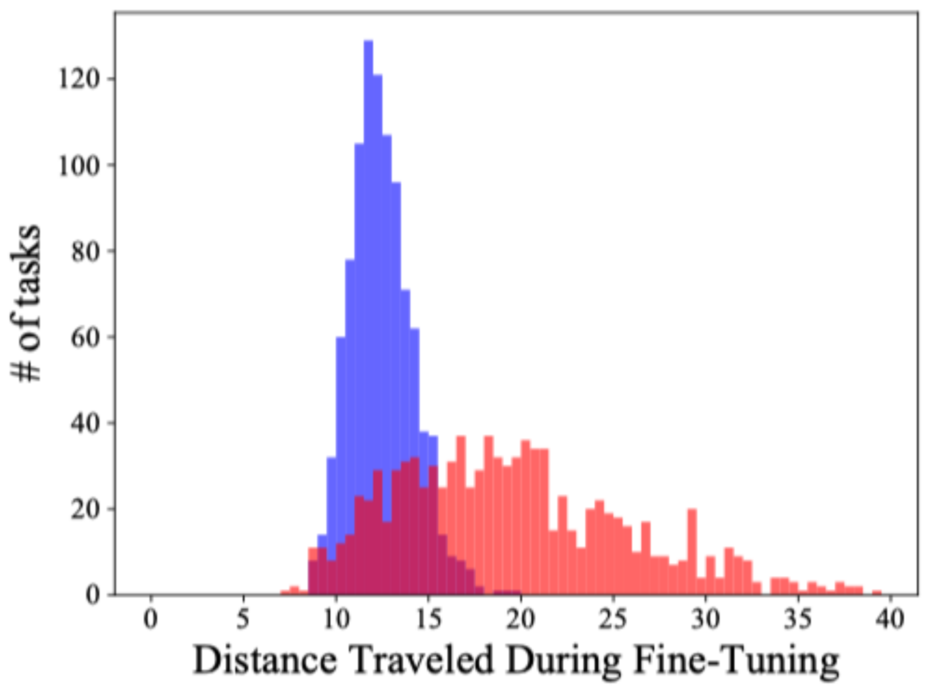
Algorithm 2 Reptile with Weight-Clustering Regularization

Require: Initial parameter vector, θ, outer learning rate, γ, inner learning rate, η, regularization coefficient, α, and distribution over tasks, $p(\mathcal{T})$.
for *meta-step* $= 1, \ldots, n$ **do**
  Sample batch of tasks, $\{\mathcal{T}_i\}_{i=1}^{m}$ from $p(\mathcal{T})$
  Initialize parameter vectors $\tilde{\theta}_i^0 = \theta$ for each task
  for $j = 1, \ldots, k$ **do**
    for $i = 1, \ldots, m$ **do**
      Calculate $\mathcal{L} = \mathcal{L}_{\mathcal{T}_i}^{j} + \alpha R_i\big(\{\tilde{\theta}_p^{j-1}\}_{p=1}^{m}\big)$
      Update $\tilde{\theta}_i^j = \tilde{\theta}_i^{j-1} - \eta\nabla_{\tilde{\theta}_i}\mathcal{L}$
    end for
  end for
  Compute difference vectors $\{g_i = \tilde{\theta}_i^k - \tilde{\theta}_i^0\}_{i=1}^{m}$
  Update $\theta \leftarrow \theta - \frac{\gamma}{m}\sum_i g_i$
end for

Experiments

Framework	1-shot	5-shot
Classical	$28.72 \pm 0.16\%$	$45.25 \pm 0.21\%$
FOMAML	$48.07 \pm 1.75\%$	$63.15 \pm 0.91\%$
Reptile	$49.97 \pm 0.32\%$	$65.99 \pm 0.58\%$
W-Clustering	$\mathbf{51.94} \pm 0.23\%$	$\mathbf{68.02} \pm 0.22\%$

Table 6. Comparison of methods on 1-shot and 5-shot mini-ImageNet 5-way classification. The top accuracy for each task is in bold. Confidence intervals have width equal to one standard error. W-Clustering denotes the Weight-Clustering regularizer.



References

1. Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta learning for fast adaptation of deep networks. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 1126–1135. JMLR. org, 2017.
2. Nichol, A. and Schulman, J. Reptile: a scalable metalearning algorithm. arXiv preprint arXiv:1803.02999, 2:2, 2018.
3. Bertinetto, L., Henriques, J. F., Torr, P. H., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. arXiv preprint arXiv:1805.08136, 2018.
4. Lee, K., Maji, S., Ravichandran, A., and Soatto, S. Metalearning with differentiable convex optimization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 10657–10665, 2019.
5. Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In Advances in Neural Information Processing Systems, pp. 4077–4087, 2017.