Unraveling Meta-Learning: Understanding Feature Representations for Few-Shot Tasks

Micah Goldblum, Steven Reich, Liam Fowl,
Renkun Ni, Valeriia Cherepanova, Tom Goldstein

University of Maryland, College Park, Maryland, USA
goldblum@umd.edu

August 14, 2020

A Brief Synopsis

What is the difference between meta-learned and classically trained networks?

- Meta-learners which fix the feature extractor during fine-tuning perform clustering in feature space.
- Improve the performance of classical training for few-shot problems by encouraging feature-space clustering.
- Relate Reptile to consensus optimization and improve its performance by enforcing a consensus penalty.

Meta-Learning for Few-Shot Classification

Require: Base model, F_θ, fine-tuning algorithm, A, learning rate, γ, and distribution over tasks, $p(\mathcal{T})$.
Initialize θ, the weights of F;
while *not done* **do**
    Sample batch of tasks, $\{\mathcal{T}_i\}_{i=1}^n$, where $\mathcal{T}_i \sim p(\mathcal{T})$ and $\mathcal{T}_i = (\mathcal{T}_i^s, \mathcal{T}_i^q)$.
    for $i = 1, \ldots, n$ **do**
        Fine-tune model on $\mathcal{T}_i$ (inner loop). New network parameters are written $\theta_i = A(\theta, \mathcal{T}_i^s)$.
        Compute gradient $g_i = \nabla_\theta \mathcal{L}(F_{\theta_i}, \mathcal{T}_i^q)$
    end for
    Update base model parameters (outer loop):
    $\theta \leftarrow \theta - \frac{\gamma}{n} \sum_i g_i$
end while

Algorithm 1: The meta-learning framework

Meta-Learning for Few-Shot Classification

- Meta-learning methods mainly differ in fine-tuning procedure.
- MAML: SGD to fine-tune all network parameters [Finn et al. 2017].
- R2-D2: Ridge regression on the one-hot labels (only fine-tune last linear layer) [Bertinetto et al. 2018].
- MetaOptNet: Differentiable solver for SVM (only fine-tune last linear layer) [Lee et al. 2019].
- ProtoNet: Nearest neighbors with class prototypes (only fine-tune last layer) [Snell et al. 2017].

Meta-Learned Feature Extractors Are Better for Few-Shot Classification

- Meta-learned models perform better than models of the same architecture trained with SGD.
- Meta-learned models are not simply well-tuned for their own fine-tuning algorithm.

Model	SVM	RR	ProtoNet	MAML
MetaOptNet-Meta	**62.64**	**60.50**	**51.99**	**55.77**
MetaOptNet-Classical	56.18	55.09	41.89	46.39
R2-D2-Meta	**51.80**	**55.89**	**47.89**	**53.72**
R2-D2-Classical	48.39	48.29	28.77	44.31

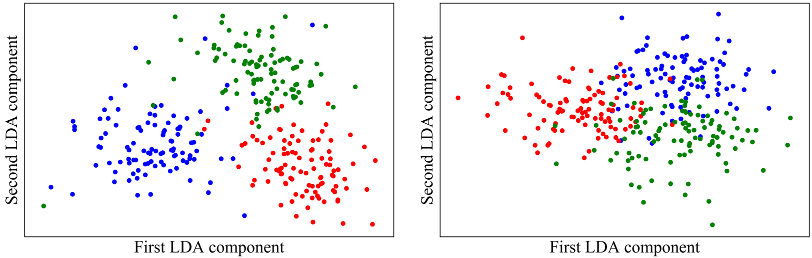
Table 1: Comparison of meta-learning and classical transfer learning models on 5-way 1-shot mini-ImageNet. Column headers denote the fine-tuning algorithm used for evaluation.

Clustering in Feature Space

Hypothesis: meta-learning algorithms **which fix the feature extractor** during the inner loop cluster each class around a point.

- Visualize feature clustering.
- Measure feature clustering.
- Sufficient condition for good few-shot classification.
- Clustering regularizers improve few-shot performance.

Visualizing Feature Clustering



(a) Meta-Learning (b) Classical Training

Figure 1: Features extracted from mini-ImageNet test data by a) ProtoNet and b) classically trained models with identical architectures.

Measuring Feature Clustering

Feature clustering: Ratio of intra-class to inter-class variance

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

Hyperplane Variation: Measures dependence of decision boundary on few-shot data sampled

$$R_{HV}(\theta, \{x_{i,j}\}) = \frac{\|(f_\theta(x_{1,1}) - f_\theta(x_{2,1})) - (f_\theta(x_{1,2}) - f_\theta(x_{2,2}))\|_2}{\|(f_\theta(x_{1,1}) - f_\theta(x_{2,1})\|_2 + \|f_\theta(x_{1,2}) - f_\theta(x_{2,2})\|_2}$$

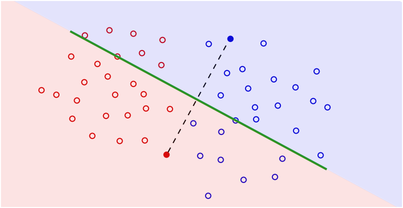
f_θ is a feature extractor with parameters θ. $x_{i,j}$ denotes sample j from class i. There are N samples in each of C classes.

Measuring Feature Clustering

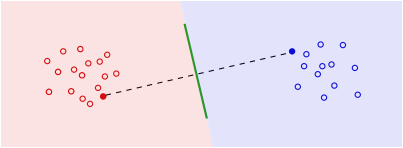
Training	Dataset	R_{FC}	R_{HV}
MetaOptNet-Meta	CIFAR-FS	**0.99**	**0.75**
MetaOptNet-Classical	CIFAR-FS	1.84	1.25
R2-D2-Meta	CIFAR-FS	**1.29**	**0.95**
R2-D2-Classical	CIFAR-FS	2.92	1.69
MetaOptNet-Meta	mini-ImageNet	**1.29**	**0.95**
MetaOptNet-Classical	mini-ImageNet	3.13	1.75
R2-D2-Meta	mini-ImageNet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

Table 2: Comparison of class separation metrics for feature extractors trained by meta-learning and classical training routines.

Why is Feature Clustering Important?



(a)



(b)

Figure 2: Both cases are linearly separable. a) Class variation is high relative to variation between classes. b) Classes move apart relative to class variation and one-shot learning yields better decision boundaries.

Feature Clustering Provably Ensures Few-Shot Performance

Theorem

Consider two random variables, X representing class 1, and Y representing class 2. Let U be the random variable equal with $P(U = X) = P(U = Y) = \frac{1}{2}$. Assume the variance ratio bound

$$\frac{\mathit{Var}[X] + \mathit{Var}[Y]}{\mathit{Var}[U]} < \epsilon$$

holds for sufficiently small $\epsilon \geq 0$. Draw random one-shot data, $x \sim X$ and $y \sim Y$, and a test point $z \sim X$. Consider linear classifier

$$c(z) = \begin{cases} 1, & \text{if } z^T(x - y) - \frac{1}{2}\|x\|^2 + \frac{1}{2}\|y\|^2 \geq 0 \\ 2, & \text{otherwise.} \end{cases}$$

This classifier correctly classifies z with probability at least $1 - \frac{32\epsilon}{1-\epsilon}$.

Feature Clustering Improves Few-Shot Classification

- Both regularizers improve few-shot classification.
- Faster than meta-learning (often by more than $10\times$).

Training	Backbone	1-shot	5-shot
R2-D2	R2-D2	65.3%	79.4%
Classical	R2-D2	62.9%	82.8%
Classical w/ R_{FC}	R2-D2	$\mathbf{65.5}\%$	$\mathbf{83.3}\%$
Classical w/ R_{HV}	R2-D2	64.6%	83.08%
MetaOptNet-SVM	MetaOptNet	72.0%	84.2%
Classical	MetaOptNet	69.5%	85.7%
Classical w/ R_{FC}	MetaOptNet	$\mathbf{72.3}\%$	$\mathbf{86.3}\%$
Classical w/ R_{HV}	MetaOptNet	72.0%	85.9%

Table 3: Comparison of methods on CIFAR-FS 5-way classification. Fine-tuning is performed with SVM except for R2-D2 in which we report numbers from the original paper.

Other Meta-Learning Methods Do Not Cluster Features

Model	R_{FC}	R_{HV}
MAML-1	3.9406	1.9434
MAML-5	3.7044	1.8901
Classical	**3.3487**	**1.8113**

Table 4: Comparison of regularizer values 1-shot and 5-shot MAML models (MAML-1 and MAML-5) as well as a classically trained model of the same architecture on mini-ImageNet training data.

An Overview of Reptile [Nichol et al. 2018]

- Inner loop - Reptile fine-tunes the whole model with gradient descent.
- Outer loop - Parameters are updated in the average direction in which parameters moved during the inner loop:

$\theta \leftarrow \theta + \frac{\gamma}{n} \sum_{i=1}^{n} (\theta_i' - \theta)$.

Reptile Performs Weight Clustering

- Reptile does not fix the feature extractor during fine-tuning.
- Reptile does not backpropagate through optimization steps.
- Reptile lacks information about the loss surface geometry when performing parameter updates.

Hypothesis: Reptile simply finds parameters that lie close to good minima for many tasks.

- Consensus formulation:

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

- Reptile almost resembles a consensus optimization method.
- But Reptile does not explicitly penalize distance from the consensus vector.

Explicit Consensus Optimization Improves Reptile

Our solution:
Explicitly minimize the quadratic penalty during the inner loop.

$$R_i\big(\{\theta_p\}_{p=1}^m\big) = d\Big(\theta_i, \frac{1}{m}\sum_{p=1}^{m}\theta_p\Big)^2,$$

where θ_p denotes current parameters on task p, and d denotes filter-normalized ℓ_2 distance between two parameter vectors.

This regularizer explicitly solves the consensus problem.

Explicit Consensus Optimization Improves Reptile

Framework	1-shot	5-shot
Classical Training	28.72%	45.25%
FOMAML	48.07%	63.15%
Reptile	49.97%	65.99%
W-Clustering	$\mathbf{51.94\%}$	$\mathbf{68.02\%}$

Table 5: Comparison of methods on 1-shot and 5-shot mini-ImageNet 5-way classification. W-Clustering denotes the Weight-Clustering regularizer.