Hybrid Consistency Training with Prototype Adaptation for Few-Shot Learning

Meng Ye Xiao Lin Giedrius Burachas Ajay Divakaran Yi Yao
SRI International
{meng.ye, xiao.lin, giedrius.burachas, ajay.divakaran, yi.yao}@sri.com

Abstract

Few-Shot Learning (FSL) aims to improve a model's generalization capability in low data regimes. Recent FSL works have made steady progress via metric learning, meta learning, representation learning, etc. However, FSL remains challenging due to the following longstanding difficulties. 1) The seen and unseen classes are disjoint, resulting in a distribution shift between training and testing. 2) During testing, labeled data of previously unseen classes is sparse, making it difficult to reliably extrapolate from labeled support examples to unlabeled query examples. To tackle the first challenge, we introduce Hybrid Consistency Training to jointly leverage two types of consistency: 1) interpolation consistency, which interpolates hidden features to imposes linear behavior locally, and 2) data augmentation consistency, which learns robust embeddings against sample variations. As for the second challenge, we use unlabeled examples to iteratively normalize features and adapt prototypes, as opposed to commonly used one-time update, for more reliable prototype-based transductive inference. We show that our method generates a 2% to 5% improvement over the state-of-the-art methods with similar backbones on five FSL datasets and, more notably, a 7% to 8% improvement for more challenging cross-domain FSL.

1. Introduction

Despite its successful applications in various computer vision tasks, deep learning still remains challenging in low data regimes. Recently, Few-Shot Learning (FSL) has drawn increasing attention in various computer vision tasks, including image classification [20, 39, 46, 49, 54], object detection [23, 24] and semantic segmentation [13, 44]. In FSL, the training classes (*i.e.*, seen or *base* classes) and the testing classes (*i.e.*, unseen or *novel* classes) are disjoint. In order to perform classification on *novel* classes using only a few labels, certain form of knowledge must be learned and transferred from *base* to *novel* classes. Such knowledge can be a

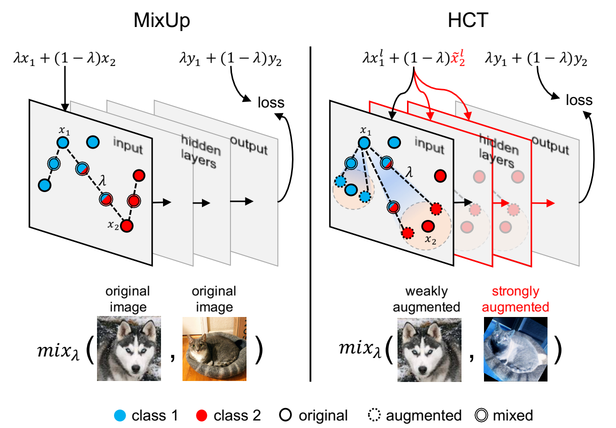

Figure 1. Comparison between Mixup [62] and Hybrid Consistency Training (HCT). Mixup imposes interpolations on lines between two examples. In HCT, the strongly augmented image is further away from the original image. Thus, interpolations cover a wider range in the input/feature space, resulting in a stronger regularization for FSL (best viewed in color).

metric space [25, 46, 54], a model initialization [15], a learning algorithm [39] , or simply an embedding model [7, 50]. While having demonstrated success on few-shot tasks, these approaches still fall short in addressing the following longstanding challenges: 1) large semantic gap between *base* and *novel* classes and 2) sparsity of labeled data of *novel* classes.

To tackle semantic gaps between *base* and *novel* classes, learning richer features to reduce overfitting on the *base* classes via incorporating knowledge learned from the images themselves is a promising direction [50]. For example, self-supervised losses, such as rotation [17] and exemplars [33], are employed in addition to the supervised loss on *base* classes for improved features [12, 19, 36]. Instead of constructing explicit surrogate tasks, another popular line of works exploit additional regularization such as consistency losses, inspired by semi-supervised learning. For example, interpolation consistency [52, 53, 62] encourages a model's local linearity and data augmentation consistency [2, 47, 58] enforces a model's local continuity.

In this paper, we propose Hybrid Consistency Training (HCT), which uniquely combines the above two consistencies by directly imposing interpolation linearity on top of weakly and strongly augmented samples across intermediate features, as opposed to commonly used post-hoc combination of two independent losses (Fig. 1). Specifically, we construct mixed features at a randomly selected network layer using a weakly and strongly augmented samples from a pair of labeled input images. The loss is measured by the cross entropy between model predictions of such mixed features and the linear combination of the ground truth labels of the original input images. Intuitively, weakly and strongly augmented samples reside in a smaller (with limited variations) and a larger (with richer variations) neighborhood of the original image, respectively. Applying interpolation consistency on strongly augmented samples enforces local continuity and linearity in a wider range, leading to richer yet more regularized embedding space. Moreover, applying interpolation consistency across intermediate features further smoothens decision boundaries throughout all network layers. Richer yet flattened (*i.e*., with fewer directions of variance) representations and smoother decision boundaries lead to improved generalization capability despite large semantic gaps.

The second challenge stems from the sparsity of labeled samples from *novel* classes. In this regard, transductive inference is introduced to leverage unlabeled data to fill in the gaps between labeled and query examples [31]. In this work, we advance prototype-based transductive inference by introducing an iterative method to calibrate features and adapt prototypes of *novel* classes using unlabeled data, referred to as Calibrated Iterative Prototype Adaptation (CIPA). While being simple, feature calibration (*e.g*., power transformation, centering, normalization) is a critical step that aligns samples from the support and query/unlabeled sets, producing an improved common ground for distance computation. Meanwhile, by estimating pseudo-labels on unlabeled data and updating prototypes iteratively, prototype estimations can be more precise despite the sparse and non-uniformly distributed labeled samples. Compared to another iterative method [22], where Sinkhorn [9] mapping is employed for pseudo labeling unlabeled data, our CIPA uses simple but effective cosine similarity, which requires much less computation. More critically, [22] assumes equal number of examples per class. In contrast, our CIPA does not rely on such assumptions and can work properly even under class imbalance.

Our contributions are:

1) We propose a Hybrid Consistency Training method built upon both interpolation and data augmentation consistencies to enforce local linearity and continuity in a wider extent (*i.e*., by incorporating strongly augmented samples) and across all network layers (*i.e*., by using Manifold Mixup). This generates significantly stronger embeddings to support generalization across large semantic gaps between the *base* and *novel* classes for improved FSL.

2) We propose an iterative prototype-based transductive inference algorithm to calibrate features and adapt class prototypes using unlabeled data. This can leverage unlabeled data to effectively fill the gaps between query and labeled samples, which are sparse and frequently non-uniformly distributed.

3) Through extensive experiments we show that our method generates a 2% to 5% improvement over the state-of-the-art (SOTA) methods with similar backbones on five FSL datasets and, more notably, a 7% to 8% improvement for more challenging cross-domain FSL (*e.g*., *mini*-ImageNet to CUB).

2. Related work

2.1. Few-shot learning

Metric learning methods learn a metric function from the base classes and use it to measure distance for novel data. Some prior work uses learnable parameters to model the metric function, for example a linear layer [25], LSTM [54] or convolutional networks [49]. Others learn a backbone network as embedding functions and use fixed metric to compute classification scores, such as euclidean distance [46], cosine similarity [6, 18] and Mahalanobis distance [1]. More recently, researchers started looking closer into image regions for calibrated metric spaces, *e.g*., [61] finds correspondences between two sets of image regions using earth mover's distance and [21] proposes a cross-attention network to focus on representative image regions.

Instead of learning a metric function, *optimization-based meta-learning* methods extract meta-knowledge from the training classes and apply it on novel data. MAML [15] learns a good model initialization that can reach optimum with a few steps of gradient descent. [39] uses LSTM as a meta-learner to learn the optimization algorithm itself that can reach convergence fast on novel classes. LEO [43] performs meta-learning using a low-dimensional space for model parameter representations.

Despite the progress in meta-learning, some recent work shows that by training a representation model on all the base classes, the resulting embeddings can be quite effective for FSL. We refer to these as *representation learning* based methods. In [7], it is shown that using distances computed on pre-trained embedding using base classes already achieve competitive results. [50] shows that learning a supervised representation from base classes followed by training a linear classifier on those representations for novel classes can also be quite effective. Tian *et al*. [50] report similar observations. Compared to complex meta-learning approaches, *representation learning* based methods are much simpler and still effective in generalizing knowledge learned from base to novel data.

Besides the methods mentioned above, another line of work incorporates self-supervised learning [4, 5, 12, 14, 19, 36] for FSL. For example, [17] finds that adding a rotation prediction task alongside the classification task to train a network leads to better FSL performance. Su *et al.* [48] note that self-supervised learning can bring greater improvements on smaller datasets or more challenging tasks. Mangla *et al.* [33] use Manifold Mixup [52] regularization as well as self-supervision loss terms (rotation [19] and exemplar [14]) to learn robust representations.

Our HCT method is based on representation learning. It is orthogonal to self-supervised techniques and can be combined with them by adding more losses in a multi-task learning manner.

2.2. Semi-supervised learning with consistency

Semi-supervised learning aims at leveraging unlabeled data in addition to labeled data to perform given tasks. Here we discuss a few semi-supervised methods using consistency-based regularization, which is closely related to our work. Virtual Adversarial Training (VAT) [35] finds local adversarial perturbations and enforces consistent model predictions despite such perturbations. FixMatch [47] is a combination of pseudo-labeling and data augmentation-based consistency regularization. For an unlabeled image, a weakly and a strongly augmented versions are generated. The weak version is used to obtain the pseudo-label for the strong version. Interpolation Consistency Training (ICT) [53] extends Mixup [62] to unlabeled data for semi-supervised learning. It uses interpolation consistency: given an interpolation of two examples as input, the model should be consistent to output the interpolation of their predictions. These consistency constrains regularize network training so that the learned networks can generalize better on test data.

Borrowing ideas from semi-supervised learning, our HCT combines interpolation and data augmentation consistency and applies these consistency-based losses on labeled data from base classes. By generating an interpolation between a weakly and a strongly augmented examples, we enforce the model output to be consistent with respect to the interpolation of their labels. We also regularize network training by not just interpolating the input images, but also interpolating the hidden features. In so doing, we introduce stronger regularization and, therefore, expect smoother manifolds.

3. Method

In FSL, it is commonly assumed that there is a training dataset $\mathcal{D}_{base}$ of base classes $\mathcal{C}_{base}$ and a test dataset $\mathcal{D}_{novel}$ of novel classes $\mathcal{C}_{novel}$. These two sets of classes are totally disjoint $\mathcal{C}_{base} \cap \mathcal{C}_{novel} = \varnothing$. Depending on different FSL approaches, the base dataset can be used either as a single dataset $\mathcal{D}_{base} = \{x_i, y_i\}_{i=1}^{N_{base}}$ (x and y denote image and label, respectively), or a source for sampling few-shot tasks (or episodes) $\mathcal{T}_{base} = \{(\mathcal{D}_i^S, \mathcal{D}_i^Q)\}_{i=1}^{N_{episode}}$, where $\mathcal{D}_i^S = \{(x_i^s, y_i^s)\}_{i=1}^{NK}$ is the support set with NK labeled examples and $\mathcal{D}_i^Q = \{(x_i^q, y_i^q)\}_{i=1}^{NQ}$ is the set with NQ query examples. This is typically referred to as an N-way K-shot problem. For evaluation, a number of novel tasks are sampled from the test dataset $\mathcal{T}_{novel} = \{(\mathcal{D}_i^S, \mathcal{D}_i^Q)\}_{i=1}^{N_{episode}}$ similarly in N-way K-shot episodes and the average accuracy on these episodes is used as the final measure of performance.

3.1. Background

Given the base and novel data, our goal is to learn an embedding network f_ϕ from *base*, so that it can be used to compute distances between *novel* images for prediction.

Prototypical Network [46] uses the centroid of support examples from each class c as its prototype. The distances between a query example and all the prototypes are computed, and then a softmax operation is applied to output the class distribution:

$$\mathbf{p}_c = \frac{1}{K} \sum_{x_i^s \in \mathcal{D}^S} \mathbb{1}_{[y_i^s = c]} f_\phi(x_i^s) \tag{1}$$

$$p_c(x^q) = \frac{\exp\Big(-\tau \cdot d\big(\mathbf{p}_c, f_\phi(x^q)\big)\Big)}{\sum_{c'} \exp\Big(-\tau \cdot d\big(\mathbf{p}_{c'}, f_\phi(x^q)\big)\Big)}, \tag{2}$$

where $d(\cdot)$ is the metric function, *e.g.*, euclidean distance or negative cosine similarity and τ is a scalar. The network is trained by minimizing a loss function defined as the cross-entropy of each query instance for all training episodes:

$$\begin{aligned}\phi^* &= \arg\min_\phi \mathop{\mathbb{E}}_{(\mathcal{D}^S, \mathcal{D}^Q) \in \mathcal{T}_{base}} \mathcal{L}_{ce}(\mathcal{D}^S, \mathcal{D}^Q) \\ &= \arg\min_\phi \mathop{\mathbb{E}}_{(\mathcal{D}^S, \mathcal{D}^Q) \in \mathcal{T}_{base}} \sum_{(x_i^q, y_i^q) \in \mathcal{D}^Q} -\log p_{y_i^q}(x_i^q)\end{aligned} \tag{3}$$

The purpose of episodic training is to simulate the few-shot evaluation protocol and reduce over-fitting on the base classes.

Classifier Baseline is a simple FSL method that learns an embedding from all base data. Just as in standard supervised learning, a fully connected layer is appended on top of f_ϕ to output logits for each base class. By sampling batches of images from $\mathcal{D}_{base}$, the embedding network can be learned by minimizing the cross entropy loss on model output and the ground truth labels:

$$p(x) = \mathsf{softmax}\big(W_{base}^T f_\phi(x) + b\big) \tag{4}$$

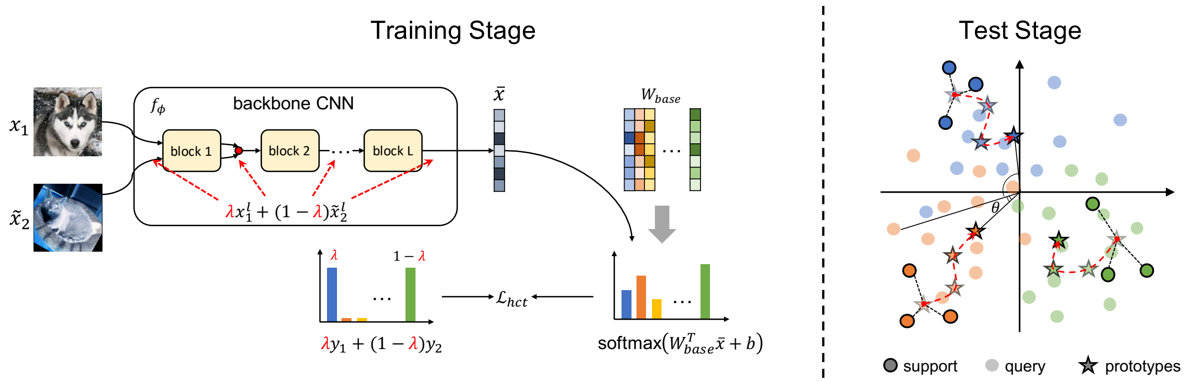

Figure 2. An overview of our approach. Left: Hybrid Consistency Training (HCT) learns an embedding network on *base* classes, with both data augmentation consistency and interpolation consistency combined into a single loss term. Right: In test stage, Calibrated Iterative Prototype Adaptation (CIPA) calibrates the features for each *novel* task and iteratively adapt prototypes to unlabeled examples.

and

$$\begin{aligned}\phi^* &= \arg\min_{\phi} \mathop{\mathbb{E}}_{(x_i,y_i)\in\mathcal{D}_{base}} \mathcal{L}_{ce}(x_i, y_i) \\ &= \arg\min_{\phi} \mathop{\mathbb{E}}_{(x_i,y_i)\in\mathcal{D}_{base}} \sum_{c=1}^{|\mathcal{C}_{base}|} -y_{i,c}\log p_c(x_i) \end{aligned} \quad (5)$$

The above two methods lay a good foundation, upon which various techniques can be added to improve FSL.

3.2. Hybrid Consistency Training

In this section, we introduce HCT, which can be viewed as a regularization technique that improves *representation learning* for few-shot task (Fig. 2 left panel). Assume that the embedding function is a composition of multiple layers $f_\phi = f^L \circ \cdots \circ f^1 \circ f^0$. The hidden representation at layer l can be obtained by passing the input image through layer $0, 1 \cdots l$: $h^l = f^l \circ \cdots \circ f^1 \circ f^0(x)$. Note that f^0 is the input layer and $h^0 = f^0(x) = x$. Given an embedding model, we optimize its weights by minimizing the following loss function

$$\mathcal{L} = \mathcal{L}_{ce} + \eta\mathcal{L}_{hct}, \quad (6)$$

where $\mathcal{L}_{ce}$ is the cross entropy loss on the base classes as in Eq. (5), η is a balancing parameter (we set it to 1 in all our experiments), and $\mathcal{L}_{hct}$ is our newly introduced hybrid consistency loss which we explain in details below.

As mentioned in Sec. 2.2, consistency training has been widely used in semi-supervised learning. In this work, we propose HCT, combining two different consistency training approaches into a unified framework to regularize model training. Given any two images x_1 and x_2, we perform weak augmentation, *e.g.* horizontal flip, to x_1 so that the augmented image is still close to the original image. We overload the notation x_1 to represent both the original image and its weakly augmented version. For x_2, we apply strong augmentation (see details in Sec. 4.1) so that it is heavily distorted and has a higher chance of being out of the local data distribution. We denote this example as $\tilde{x}_2$. To generate an interpolation between x_1 and $\tilde{x}_2$, we feed them both into the embedding network and then randomly choose a layer l to get their hidden representations:

$$\begin{aligned} x_1^l &= f^l \circ \cdots f^1 \circ f^0(x_1) \\ \tilde{x}_2^l &= f^l \circ \cdots f^1 \circ f^0(\tilde{x}_2). \end{aligned} \quad (7)$$

The hidden representations are mixed and passed through remaining layers to get the final feature representation $\bar{x}$:

$$\begin{aligned} \bar{x}^l &= \lambda \cdot x_1^l + (1-\lambda)\cdot \tilde{x}_2^l \\ \bar{x} &= f^L \circ \cdots f^{l+1}(\bar{x}^l). \end{aligned} \quad (8)$$

The corresponding target $\bar{y}$ is the interpolation of the ground truth one-hot label vectors y_1 and y_2 of the original input samples x_1 and x_2:

$$\bar{y} = \lambda \cdot y_1 + (1-\lambda)\cdot y_2. \quad (9)$$

Then, the loss function on these interpolated examples is

$$\mathcal{L}_{hct} = \mathop{\mathbb{E}}_{\substack{(x_1,y_1)\in\mathcal{D}_{base} \\ (x_2,y_2)\in\mathcal{D}_{base} \\ \lambda\sim Beta(\alpha,\alpha),\ l\sim U(0,L)}} \sum_{c=1}^{|\mathcal{C}_{base}|} -\bar{y}_c \log p_c(\bar{x}). \quad (10)$$

HCT combines interpolation consistency [53, 62] and data augmentation consistency [2, 47, 58] in a unique and tightly integrated way: the generated new data points not only cover linear space between examples, but also expand further to the regions where heavily distorted examples reside. By doing this at a random layer each time, hidden representations at all levels are regularized. This leads to a smoother manifold that generalizes better to novel classes. HCT can also be combined with other representation learning techniques, *e.g.*, self-supervised rotation classification $\mathcal{L}_{rot}$, by simply adding another head and performing multi-task learning (denoted as HCT_R), which often results in further improved representations.

3.3. Calibrated Iterative Prototype Adaptation

We use the embedding model f_{ϕ^*} trained by HCT to infer predictions for novel data. Given a novel task $\mathcal{T}_{novel}^{(i)} = (\mathcal{D}_i^S, \mathcal{D}_i^Q)$, we first extract features of both the support examples and the query examples. A straightforward way to get class probabilities is to compute class prototypes and then the distances between query examples and each prototype followed by softmax, as in Eq. (1).

However, due to the sparsity and sporadicity (*i.e.*, non-uniformly distributed) of the support examples, the quality of prototypes varies substantially from episode to episode. In order to better estimate class prototypes as well as better adapt to specific tasks, we need to make full use of unlabeled query examples for semi-supervised or transductive inference. As described in [40], pseudo-labels obtained by Eq. (2) can be used to update prototypes in a K-means step:

$$\tilde{\mathbf{p}}_c = \frac{\sum\limits_{(x_i^s, y_i^s) \in \mathcal{D}^S} \mathbb{1}_{[y_i^s = c]} f_{\phi^*}(x_i^s) + \sum\limits_{x_j^q \in \mathcal{D}^Q} p_c(x_j^q) f_{\phi^*}(x_j^q)}{\sum\limits_{(x_i^s, y_i^s) \in \mathcal{D}^S} \mathbb{1}_{[y_i^s = c]} + \sum\limits_{x_j^q \in \mathcal{D}^Q} p_c(x_j^q)} \tag{11}$$

Another problem of centroid-nearest neighbor method is that, since only a few data points are sampled, the data distribution of tasks vary heavily from each other. Thus, certain transformations [22, 56] are needed to calibrate the features. To this end, we propose CIPA that: 1) calibrates the features for better distance computation, and 2) iteratively predicts pseudo-labels on unlabeled data and updates the estimation of prototypes progressively (Fig. 2 right panel). The inference procedure is shown in Algorithm 1.

In our experiments, we have found that this straightforward iterative inference algorithm can greatly improve FSL performance when unlabeled data is available. Hu *et al.* [22] also uses an iterative approach to update class centers. However, they assume that the test set has an equal number of examples for each class and use Sinkhorn mapping [9] to find the best match. While improved FSL performace is demonstrated, this is, to certain degree, due to the fact that episodes constructed under the evaluation protocols of FSL datasets do have uniform class distribution. Their method, therefore, may find it difficult in dealing with imbalanced classes. Our CIPA does not rely on such assumptions and will work properly under class imbalance, which is critical for real-world applications.

In Algorithm 1, we use query examples to update the prototypes. However, CIPA is not limited to such a transductive setting and can be extended to semi-supervised FSL, where another auxiliary set of unlabeled data is used instead of query examples themselves. We have conducted experiments and verified the effectiveness of CIPA for semi-supervised FSL (see the supplementary materials).

Algorithm 1: Calibrated Iterative Prototype Adaptation (CIPA)

Input: $\mathcal{T}_{novel}^{(i)} = (\mathcal{D}^S, \mathcal{D}^Q)$, f_{ϕ^*}, N_{iter}, σ, τ
Output: $\hat{p}(x_i^q)$ for each $x_i^q \in \mathcal{D}^Q$

```
/* power transformation */
```
$x_i^s \leftarrow \frac{(x_i^s)^\beta}{\|(x_i^s)^\beta\|}$, $x_i^q \leftarrow \frac{(x_i^q)^\beta}{\|(x_i^q)^\beta\|}$
```
/* zero-mean */
```
$x_i^s \leftarrow x_i^s - \frac{1}{NK}\sum_{x_i^s \in \mathcal{D}^S} x_i^s$
$x_i^q \leftarrow x_i^q - \frac{1}{NQ}\sum_{x_i^q \in \mathcal{D}^Q} x_i^q$
```
/* l2 normalization */
```
$x_i^s \leftarrow \frac{x_i^s}{\|x_i^s\|}$, $x_i^q \leftarrow \frac{x_i^q}{\|x_i^q\|}$
```
/* compute and update prototypes */
```
Initialize $\mathbf{p}^{(0)}$ using Eq. (1).
for $t = 1, 2, \ldots N_{iter}$ **do**
  $p^{(t)}(x^q) \leftarrow \mathsf{softmax}(\tau \cdot cos(x^q, \mathbf{p}^{(t-1)}))$
  Compute new $\tilde{\mathbf{p}}^{(t)}$ using Eq. (11) and $p^{(t)}(x^q)$
  $\mathbf{p}^{(t)} \leftarrow \sigma\tilde{\mathbf{p}}^{(t)} + (1-\sigma)\mathbf{p}^{(t-1)}$
```
/* predict using the final prototypes */
```
$\hat{p}(x^q) \leftarrow \mathsf{softmax}(\tau \cdot cos(x^q, \mathbf{p}^{(N_{iter})}))$
return $\hat{p}(x^q)$

4. Experiments

4.1. Settings

Datasets. We conducted experiments on five FSL datasets: 1) ***mini*-ImageNet** [54] is derived from the ILSVRC2012 [42] dataset. It contains 100 randomly sampled classes and is split into 64, 16 and 20 classes for train, validation and test, respectively. Each class has 600 images. 2) ***tiered*-ImageNet** [40] is also a subset of ILSVRC2012 [42]. It contains in total 34 super categories and is split into 20, 6 and 8 for train, validation and test, respectively. The corresponding class numbers are 351, 97 and 160. On average, each class has around 1280 images. 3) **CIFAR_FS** [3] is a few-shot learning dataset that contains all 100 classes from CIFAR100 [26]. The dataset is randomly split into 64, 16 and 20 classes for train, validation and test. Each class has 600 images. 4) **FC_100** [3] is also derived from CIFAR100 [26]. But it is instead split into 60, 20 and 20 classes that are from 12, 4 and 4 super categories. 5) **CUB** [55] is a dataset of 200 fine-grained bird species. We follow [6] to split the dataset into 100, 50 and 50 for train, validation and test and each class has around 59 images. For all these five datasets, we resize the images into 84×84 if they are not so already.

Training settings. In all our experiments, we use ResNet-12 [7] as our backbone network. To train the network, we use the Adam optimizer with a learning rate of 0.001

Setting	Method	Backbone	**5-way *mini*-ImageNet** 1-shot	5-shot	**5-way *tiered*-ImageNet** 1-shot	5-shot
In.	TADAM [37]	ResNet12	58.50 ± 0.30	76.70 ± 0.30	–	–
	ProtoNet [46]†	ResNet12	59.25 ± 0.64	75.60 ± 0.48	61.74 ± 0.77	80.00 ± 0.55
	MetaOptNet-SVM [27]	ResNet12	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
	SNAIL [34]	ResNet15	55.71 ± 0.99	68.88 ± 0.92	–	–
	SimpleShot [56]	ResNet18	62.85 ± 0.20	80.02 ± 0.14	69.09 ± 0.22	84.58 ± 0.16
	DeepEMD [61]	ResNet12	65.91 ± 0.82	82.41 ± 0.56	71.16 ± 0.87	86.03 ± 0.58
	LEO [43]	WRN-28-10	61.76 ± 0.08	77.59 ± 0.12	66.33 ± 0.05	81.44 ± 0.09
	CC+rot [17]	WRN-28-10	62.93 ± 0.45	79.87 ± 0.33	70.53 ± 0.51	84.98 ± 0.36
	$S2M2_R$ [33]	WRN-28-10	64.93 ± 0.18	83.18 ± 0.11	73.71 ± 0.22	**88.59 ± 0.14**
Trans.	TPN [31]	ResNet12	59.46	75.65	58.68§	74.26§
	Trans. Fine-Tuning [11]	ResNet12	62.35 ± 0.66	74.53 ± 0.54	68.41 ± 0.73	83.41 ± 0.52
	TEAM [38]	ResNet18	60.07	75.90	–	–
	LR + ICI [57]	ResNet12	66.80	79.26	80.79	**87.92**
	DSN-MR [45]	ResNet12	64.60 ± 0.72	79.51 ± 0.50	67.39 ± 0.82	82.85 ± 0.56
	EPNet [41]	ResNet12	66.50 ± 0.89	81.06 ± 0.60	76.53 ± 0.87	**87.32 ± 0.64**
	FEAT [59]	ResNet18	66.78 ± 0.20	82.05 ± 0.14	70.80 ± 0.23	84.79 ± 0.16
	LaplacianShot [63]	ResNet18	72.11 ± 0.19	82.31 ± 0.14	78.98 ± 0.21	86.39 ± 0.16
	HCT_R + CIPA (ours)	ResNet12	**76.94 ± 0.24**	**85.10 ± 0.14**	**81.70 ± 0.25**	**87.91 ± 0.15**
	ICA+MSP [29]	DenseNet	77.06 ± 0.26	84.99 ± 0.14	84.29 ± 0.25	89.31 ± 0.15
	PT+MAP [22]	WRN◇/DenseNet*	82.92 ± 0.26◇	88.82 ± 0.13◇	85.67 ± 0.26*	90.45 ± 0.14*

Table 1. Results on *mini*-ImageNet and *tiered*-ImageNet. In. and Trans. stand for inductive and transductive, respectively. Methods marked with † are reported in Lee *et al*. [27], while those with § are from Wang *et al*. [57]. Our accuracies are averaged over 10k episodes.

and train for 300 epochs (60 on *tiered*-ImageNet). During the first 1/3 of total epochs we use $\mathcal{L}_{ce} + \mathcal{L}_{rot}$, for the remaining 2/3 of the epochs we add the $\mathcal{L}_{hct}$ loss term. To interpolate examples, by default we use $\alpha = 2$ to sample $\lambda \sim Beta(\alpha, \alpha)$ unless stated otherwise. For the weak augmentation, we use random crop and random flip at 50% chance. For the strong augmentation, we follow FixMatch [47] and use RandAugment [8]. Each time, 2 out of 14 augmentations are randomly selected and applied to the image, after which a random square region in the image is cut out [10]. We use the same settings for all datasets to obtain our main results. Performance on validation data is monitored during training for model selection.

Evaluation settings. In the test phase, we fix the trained backbone network and use it as a feature extractor. The extracted features of the support and query samples are used by CIPA to predict their classes. We use $\beta = 0.5$, $\sigma = 0.2$, $\tau = 10$ and $N_{iter} = 20$ for all experiments. In each experiment, a number of novel episodes (600 or 10,000) are sampled. Each episode contains N classes, and each class has K support and 15 query examples. Note that we do not use any auxiliary unlabeled examples as did in semi-supervised FSL [28, 60] and thus these methods are not comparable to ours. We report the average accuracy and 95% confidence interval as performance measurements.

4.2. Main results

Standard few-shot learning. We separate comparison methods into the inductive and transductive groups. In both groups, we list SOTA results with a similar backbone (*e.g.*, ResNet12, ResNet 18) as well as those with heavier backbones (*e.g.*, WRN and DenseNet). Note that performance achieved with deeper backbones are not directly comparable to our results, those are listed just for reference.

We summarize the results on *mini*-ImageNet and *tiered*-ImageNet in Tab. 1. Our method, HCT_R+ CIPA, has achieved the best performance among all comparison methods on the *mini*-ImageNet dataset. Comparing to LaplacianShot [63], the best performing method reported using a ResNet18 backbone, we achieve more than 4% and nearly 3% improvements on 1-shot and 5-shot, respectively. For *tiered*-ImageNet, HCT_R+CIPA yields the best performance on 1-shot while being on par with EPNet [41] and LR+ICI [57] on 5-shot. The results on CIFAR_FS and FC100 are summarized in Tab. 2. Similarly, our method achieves the best performance across all settings. Note that some of the methods in the inductive group, such as CC+rot and $S2M2_R$, use a larger network (*e.g.*, WRN-28-10). Our method still outperforms them, showing that our training method combined with transductive inference can compensate for the disadvantages of using a lighter network.

Results on CUB, a dataset of fine-grained bird species, are summarized in Tab. 3. Again, our approach has achieved the best performance on both 1-shot and 5-shot with an improvement of ~5% and ~2%, respectively, over LR+ICI [57], the best reported method in literature using a ResNet12 backbone. Notably, even comparing to transductive methods with a larger backbone of WRN-28-10 (*e.g.*,

Setting	Method	Backbone	5-way CIFAR_FS		5-way FC100	
			1-shot	5-shot	1-shot	5-shot
In.	ProtoNet [46]†	ResNet12	72.2 ± 0.7	83.5 ± 0.5	37.5 ± 0.6	52.5 ± 0.6
	MetaOptNet-SVM [27]	ResNet12	72.0 ± 0.7	84.2 ± 0.5	41.1 ± 0.6	55.5 ± 0.6
	TADAM [37]	ResNet12	–	–	40.1 ± 0.4	56.1 ± 0.4
	SimpleShot [56]	ResNet10	–	–	40.13 ± 0.18	53.63 ± 0.18
	DeepEMD [61]	ResNet12	–	–	46.47 ± 0.78	63.22 ± 0.71
	CC+rot [17]	WRN-28-10	76.09 ± 0.30	87.83 ± 0.21	–	–
	S2M2$_R$ [33]	WRN-28-10	74.81 ± 0.19	87.47 ± 0.13	–	–
Trans.	TPN [31]	ResNet12	65.89§	79.38§	–	–
	TEAM [38]	ResNet18	70.43	81.25	–	–
	Transductive Fine-Tuning [11]	ResNet12	70.76 ± 0.74	81.56 ± 0.53	41.89 ± 0.59	54.96 ± 0.55
	LR + ICI [57]	ResNet12	73.97	84.13	–	–
	DSN-MR [45]	ResNet12	75.6 ± 0.9	86.2 ± 0.6	–	–
	HCT$_R$ + CIPA (ours)	ResNet12	**85.72** ± **0.21**	**89.69** ± **0.14**	**53.30** ± **0.25**	**64.90** ± **0.20**
	PT+MAP [22]	WRN-28-10	87.69 ± 0.23	90.68 ± 0.15	–	–

Table 2. Results on CIFAR_FS and FC100. Our accuracies are averaged over 10k episodes.

Method	Backbone	5-way CUB	
		1-shot	5-shot
DeepEMD [61]	ResNet12	75.65 ± 0.83	88.69 ± 0.50
S2M2$_R$ [33]	WRN-28-10	80.68 ± 0.81	90.85 ± 0.44
TEAM [38]	ResNet18	80.16	87.17
LaplacianShot [63]	ResNet18	80.96	88.68
LR+ICI [57]	ResNet12	88.06	92.53
HCT$_R$ + CIPA (ours)	ResNet12	**93.03** ± **0.15**	**94.90** ± **0.08**
BD-CSPN [30]	WRN-28-10	87.45	91.74
PT+MAP [22]	WRN-28-10	91.55 ± 0.19	93.99 ± 0.10

Table 3. Results on CUB. Ours are averaged over 10k episodes.

Method	Backbone	*mini*-ImageNet→ CUB	
		1-shot	5-shot
Mat. Net [54] + FT†	ResNet10	36.61 ± 0.53	55.23 ± 0.83
Rel. Net [49] + FT†	ResNet10	44.07 ± 0.77	59.46 ± 0.71
S2M2$_R$ [33]	WRN-28-10	48.24 ± 0.84	70.44 ± 0.75
GNN [16] + FT†	ResNet10	47.47 ± 0.75	66.98 ± 0.68
LaplacianShot [63]	ResNet18	55.46	66.33
HCT$_R$ + CIPA (ours)	ResNet12	**62.15** ± **1.08**	**74.25** ± **0.77**
PT+MAP [22]	WRN-28-10	62.49 ± 0.32	76.51 ± 0.18

Table 4. Results for cross-domain FSL. Our accuracies are averaged over 600 episodes. † are reported in Tseng *et al.* [51].

PT+MAP [22]) , our method still remains the best. The results on CUB strongly suggest that regularizing learned embedding in a wider extent and across network layers can help to learn rich and robust representations to significantly benefit FSL on fine-grained classes.

Cross-domain FSL. To study the robustness of representations learned via our HCT across datasets with certain amounts of covariate shift, we evaluate its performance under cross-domain scenarios as an outreaching test. We train models on *mini*-ImageNet and test them on CUB. From Tab. 4, our method achieves the best performance on both 1-shot and 5-shot tasks, with an improvement of 7% and 8% over LaplacianShot, respectively. The 1-shot accuracy is on par with PT+MAP despite that HCT uses a shallower network. This manifests that our method not only works under in-domain settings, but also can generalize well under the more challenging cross-domain settings.

4.3. Ablation studies

HCT for embedding learning. To better understand how each component of HCT affects the learning of representations, we design our experiments in two directions (Tab. 5): 1) how the embedding is trained (row-wise) and 2) what inference algorithm is used (column-wise). To train an embedding model, we start with "Classifier Baseline", which only uses $\mathcal{L}_{ce}$. We then add $\mathcal{L}_{mm}$ for Manifold Mixup, or $\mathcal{L}_{hct}$ for our HCT. Beyond these, adding another rotation loss $\mathcal{L}_{rot}$ yields S2M2$_R$ and HCT$_R$. As for inference, we compare our CIPA against ProtoNet [46], a centroid-nearest neighbor based method, and SemiPN [40], an extension of ProtoNet that makes use of unlabeled data.

From Tab. 5, we have several observations: 1) Comparing three inference algorithms, our CIPA is consistently the best across all experiments. 2) Comparing Classifier Baseline and HCT, adding $\mathcal{L}_{hct}$ leads to 2~3% improvements across all inference algorithms on *mini*-ImageNet. On CUB, the improvements are marginal in inductive settings (*i.e.*, using PN), but more noticeable in transductive settings (*i.e.*, using SemiPN or CIPA). 3) Comparing S2M2$_R$ and HCT$_R$, *i.e.*, $\mathcal{L}_{mm}$ v.s. $\mathcal{L}_{hct}$ on top of $\mathcal{L}_{ce} + \mathcal{L}_{rot}$, HCT$_R$ is significantly better than S2M2$_R$ on CUB while being on par with it on *mini*-ImageNet. For a more thorough comparison between Manifold Mixup and HCT on different α values, please see details in the supplementary materials.

Based on these observations, we conclude that the benefit introduced by HCT depends on: 1) the method onto which HCT is added and 2) the dataset to which HCT is applied. Overall, HCT brings improvements for FSL, espe-

Method	Train				PN		SemiPN		CIPA	
	$\mathcal{L}_{ce}$	$\mathcal{L}_{mm}$	$\mathcal{L}_{hct}$	$\mathcal{L}_{rot}$	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
***mini*-ImageNet**										
Classifier Baseline	✓				56.48	75.62	66.14	77.72	70.84	80.59
Manifold Mixup	✓	✓			57.07	78.09	68.25	80.12	73.69	83.06
HCT	✓		✓		58.54	78.43	69.38	80.33	74.74	82.91
S2M2_R	✓	✓		✓	59.66	77.60	68.77	79.99	76.54	85.16
HCT_R	✓		✓	✓	60.33	77.66	69.38	80.32	77.26	84.89
CUB										
Classifier Baseline	✓				67.56	85.63	79.57	87.95	84.96	89.84
Manifold Mixup	✓	✓			65.78	86.53	79.26	88.84	86.12	90.95
HCT	✓		✓		67.90	86.73	80.43	89.20	86.79	91.02
S2M2_R	✓	✓		✓	73.84	88.26	83.52	90.05	88.40	91.93
HCT_R	✓		✓	✓	81.68	92.39	89.47	93.22	93.27	94.77

Table 5. Ablation study on HCT. Accuracies are averaged over 600 episodes.

	***mini*-ImageNet**						
	center	l_2 norm.	pow.	σ	N_{iter}	1-shot	5-shot
(a)				N/A	0	60.33	77.66
(b)	✓			N/A	0	63.48	78.39
(c)	✓	✓		N/A	0	63.48	78.73
(d)	✓	✓	✓	N/A	0	65.96	81.35
(e)	✓	✓	✓	1.0	1	72.92	83.94
(f)	✓	✓	✓	1.0	20	**78.19**	84.74
(g)	✓	✓	✓	0.2	20	77.26	**84.89**

Table 6. Ablation study on CIPA. Accuracies are averaged over 600 episodes on *mini*-ImageNet.

cially when used in combination with our CIPA.

CIPA for transductive inference. We then study how each component of CIPA affects the final performance in Tab. 6. Comparing rows (a) and (b), we find that simply subtracting the mean induces a nearly ~3% improvement on 1-shot. This indicates the existence of shift between the data distributions of few-shot tasks and the true data distribution and a simple centering can effectively compensate for such a shift. Comparing (c) and (d), we note that power transform [22] also introduces an improvement of ~2% for both 1- and 5-shot. As expected, the greatest increase, 7% on 1-shot and ~3% on 5-shot, comes from adapting the prototypes using unlabeled examples ((e) vs. (d)). Tuning the adaptation parameters also helps improving the performance of CIPA ((e) to (g)). For a more detailed parameter sensitivity analysis please see the supplementary materials. Finally we plot a randomly sampled task in Fig. 3 for an intuitive understanding of the inner workings and effect of CIPA. We can see that, initial prototypes are those 1-shot labeled examples, which are sub-optimal for prediction. Through iterative adaptation, those prototypes gradually move toward the center of each class, generating better predictions.

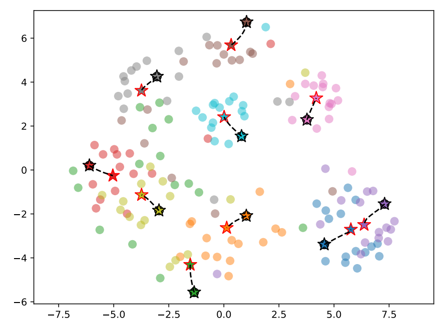

Figure 3. Visualization of a 10-way 1-shot task from *mini*-ImageNet using t-SNE [32]. Colors represent ground truth labels. "☆" with black outline are initial prototypes and those with red outline are adapted prototypes. Dashed lines show how they gradually evolve. Circles with no outline are unlabeled query examples and those with black outline are labeled support examples (overlaied with initial prototypes).

5. Conclusion

Few-Shot Learning is a critical problem to be addressed for a wider utilization of deep learning. In this paper, We tackled two longstanding difficulties in FSL. 1) To generalize from base to novel classes, we proposed hybrid consistency training (HCT), a combination of interpolation consistency and data augmentation consistency to regularize the learning of representations. 2) To bridge the gap between sparse support and query examples, we developed a transductive inference algorithm, *i.e.*, CIPA, to calibrate features and adapt prototypes iteratively. Through extensive experiments, we have shown that our method can achieve SOTA performance on all five FSL datasets. Ablation studies also justified the necessity and quantified the effectiveness of each component in HCT and CIPA.

6. Acknowledgments

This material is based upon work supported by the United States Air Force under Contract No. FA850-19-C-0511. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.

References

[1] Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *CVPR*, 2020. 2

[2] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *ICLR*, 2020. 1, 4

[3] Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *ICLR*, 2019. 5

[4] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018. 3

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 3

[6] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *ICLR*, 2019. 2, 5

[7] Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning. *arXiv preprint arXiv:2003.04390*, 2020. 1, 2, 5

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR Workshops*, 2020. 6

[9] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013. 2, 5

[10] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 6

[11] Guneet S Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *ICLR*, 2020. 6, 7

[12] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 1, 3

[13] Nanqing Dong and Eric P Xing. Few-shot semantic segmentation with prototype learning. In *BMVC*, 2018. 1

[14] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *NeurIPS*, 2014. 3

[15] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 1, 2

[16] Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. In *ICLR*, 2018. 7

[17] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *ICCV*, 2019. 1, 3, 6, 7

[18] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018. 2

[19] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 1, 3

[20] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *ICCV*, 2017. 1

[21] Ruibing Hou, Hong Chang, MA Bingpeng, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *NeurIPS*, 2019. 2

[22] Yuqing Hu, Vincent Gripon, and Stéphane Pateux. Leveraging the feature distribution in transfer-based few-shot learning. *arXiv preprint arXiv:2006.03806*, 2020. 2, 5, 6, 7, 8

[23] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *ICCV*, 2019. 1

[24] Leonid Karlinsky, Joseph Shtok, Sivan Harary, Eli Schwartz, Amit Aides, Rogerio Feris, Raja Giryes, and Alex M Bronstein. Repmet: Representative-based metric learning for classification and few-shot object detection. In *CVPR*, 2019. 1

[25] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning Workshop*, 2015. 1, 2

[26] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 5

[27] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019. 6, 7

[28] Xinzhe Li, Qianru Sun, Yaoyao Liu, Shibao Zheng, Qin Zhou, Tat-Seng Chua, and Bernt Schiele. Learning to self-train for semi-supervised few-shot classification. In *NeurIPS*, 2019. 6

[29] Moshe Lichtenstein, Prasanna Sattigeri, Rogerio Feris, Raja Giryes, and Leonid Karlinsky. Tafssl: Task-adaptive feature sub-space learning for few-shot classification. In *ECCV*, 2020. 6

[30] Jinlu Liu, Liang Song, and Yongqiang Qin. Prototype rectification for few-shot learning. *arXiv preprint arXiv:1911.10713*, 2019. 7

[31] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *ICLR*, 2019. 2, 6, 7

[32] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 8

[33] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *WACV*, 2020. 1, 3, 6, 7

[34] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *ICLR*, 2018. 6

[35] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE TPAMI*, 41(8):1979–1993, 2018. 3

[36] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016. 1, 3

[37] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, 2018. 6, 7

[38] Limeng Qiao, Yemin Shi, Jia Li, Yaowei Wang, Tiejun Huang, and Yonghong Tian. Transductive episodic-wise adaptive metric for few-shot learning. In *ICCV*, 2019. 6, 7

[39] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 1, 2

[40] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 5, 7

[41] Pau Rodriguez, Issam Laradji, Alexandre Drouni, and Alexandre Lacoste. Embedding propagation: smoother manifold for few-shot classification. In *ECCV*, 2020. 6

[42] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 5

[43] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *ICLR*, 2019. 2, 6

[44] Mennatullah Siam, Boris N Oreshkin, and Martin Jagersand. Amp: Adaptive masked proxies for few-shot segmentation. In *ICCV*, 2019. 1

[45] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *CVPR*, 2020. 6, 7

[46] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 1, 2, 3, 6, 7

[47] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. 1, 3, 4, 6

[48] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. When does self-supervision improve few-shot learning? In *ECCV*, 2020. 3

[49] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018. 1, 2, 7

[50] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *arXiv preprint arXiv:2003.11539*, 2020. 1, 2

[51] Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. In *ICLR*, 2020. 7

[52] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019. 1, 3

[53] Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. In *IJCAI*, 2019. 1, 3, 4

[54] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016. 1, 2, 5, 7

[55] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 5

[56] Yan Wang, Wei-Lun Chao, Kilian Q Weinberger, and Laurens van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*, 2019. 5, 6, 7

[57] Yikai Wang, Chengming Xu, Chen Liu, Li Zhang, and Yanwei Fu. Instance credibility inference for few-shot learning. In *CVPR*, 2020. 6, 7

[58] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019. 1, 4

[59] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *CVPR*, 2020. 6

[60] Zhongjie Yu, Lin Chen, Zhongwei Cheng, and Jiebo Luo. Transmatch: A transfer-learning scheme for semi-supervised few-shot learning. In *CVPR*, 2020. 6

[61] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *CVPR*, 2020. 2, 6, 7

[62] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 1, 3, 4

[63] Imtiaz Masud Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. In *ICML*, 2020. 6, 7