Rethinking Few-Shot Image Classification: a Good Embedding Is All You Need?

Yonglong Tian[1*] Yue Wang[1*] Dilip Krishnan[2] Joshua B. Tenenbaum[1] Phillip Isola[1]
[1]MIT CSAIL [2]Google Research

Abstract

The focus of recent meta-learning research has been on the development of learning algorithms that can quickly adapt to test time tasks with limited data and low computational cost. Few-shot learning is widely used as one of the standard benchmarks in meta-learning. In this work, we show that a simple baseline: learning a supervised or self-supervised representation on the meta-training set, followed by training a linear classifier on top of this representation, outperforms state-of-the-art few-shot learning methods. An additional boost can be achieved through the use of self-distillation. This demonstrates that using a good learned embedding model can be more effective than sophisticated meta-learning algorithms. We believe that our findings motivate a rethinking of few-shot image classification benchmarks and the associated role of meta-learning algorithms. Code is available at: http://github.com/WangYueFt/rfs/.

1. Introduction

Few-shot learning measures a model's ability to quickly adapt to new environments and tasks. This is a challenging problem because only limited data is available to adapt the model. Recently, significant advances [55, 54, 52, 12, 46, 48, 56, 34, 44, 60, 26, 28] have been made to tackle this problem using the ideas of meta-learning or "learning to learn".

Meta-learning defines a family of tasks, divided into disjoint meta-training and meta-testing sets. Each task consists of limited training data, which requires fast adaptability [45] of the learner (e.g., the deep network that is fine-tuned). During meta-training/testing, the learner is trained and evaluated on a task sampled from the task distribution. The performance of the learner is evaluated by the average test accuracy across many meta-testing tasks. Methods to tackle this problem can be cast into two main categories: optimization-based methods and metric-based methods. Optimization-based methods focus on designing algorithms that can quickly adapt to each task; while metric-based methods aim to find good metrics (usually kernel functions) to side-step the need for inner-loop optimization for each task.

Meta-learning is evaluated on a number of domains such as few-shot classification and meta-reinforcement learning. Focusing on few-shot classification tasks, a question that has been raised in recent work is whether it is the meta-learning algorithm or the learned representation that is responsible for the fast adaption to test time tasks. [39] suggested that feature reuse is main factor for fast adaptation. Recently, [9] proposed transductive fine-tuning as a strong baseline for few-shot classification; and even in a regular, inductive, few-shot setup, they showed that fine-tuning is only slightly worse than state-of-the-art algorithms. In this setting, they fine-tuned the network on the meta-testing set and *used* information from the testing data. Besides, [5] shows an improved fine-tuning model performs slightly worse than meta-learning algorithms.

In this paper, we propose an extremely simple baseline that suggests that good learned representations are more powerful for few-shot classification tasks than the current crop of complicated meta-learning algorithms. Our baseline consists of a *linear* model learned on top of a pre-trained embedding. Surprisingly, we find this outperforms *all other meta-learning algorithms* on few-shot classification tasks, often by large margins. The differences between our approach and that of [9] are: we *do not* utilize information from testing data (since we believe that inductive learning is more generally applicable to few-shot learning); and we use a fixed neural network for feature extraction, rather than fine-tuning it on the meta-testing set. The findings in concurrent works [6, 20] are inline with our simple baseline.

Our model learns representations by training a neural network on the entire meta-training set: we merge all meta-training data into a single task and a neural network is asked to perform either ordinary classification or self-supervised learning, on this combined dataset. The classification task is equivalent to the pre-training phase of TADAM [34] and LEO [44]. After training, we keep the pre-trained network up to the penultimate layer and use it as a feature extractor. During meta-testing, for each task, we fit a linear classifier on the features extracted by the pre-trained network. In contrast to [9] and [39], we *do not* fine-tune the neural network.

*: equal contribution.

Furthermore, we show that self-distillation on this baseline provides an additional boost. Self-distillation is a form of knowledge distillation [18], where the student and teacher models are *identical* in architecture and task. We apply self-distillation to the pre-trained network.

Contributions. Our key contributions are:

- A surprisingly simple baseline for few-shot learning, which achieves the state-of-the-art. This baseline suggests that many recent meta-learning algorithms are *no better* than simply learning a good representation through a proxy task, e.g., image classification.
- Building upon the simple baseline, we use self-distillation to further improve performance.
- Our combined method achieves an average of 3% improvement over the previous state-of-the-art.
- Beyond supervised training, we show that representations learned with state-of-the-art self-supervised methods achieve similar performance as fully supervised methods. Thus we can "learn to learn" simply by learning a good self-supervised embedding.

2. Related works

Metric-based meta-learning. The core idea in metric-based meta-learning is related to nearest neighbor algorithms and kernel density estimation. Metric-based methods embed input data into fixed dimensional vectors and use them to design proper kernel functions. The predicted label of a query is the weighted sum of labels over support samples. Metric-based meta-learning aims to learn a task-dependent metric. [22] used Siamese network to encode image pairs and predict confidence scores for each pair. Matching Networks [54] employed two networks for query samples and support samples respectively and used an LSTM with read-attention to encode a full context embedding of support samples. Prototypical Networks [46] learned to encode query samples and support samples into a shared embedding space; the metric used to classify query samples is the distance to prototype representations of each class. Instead of using distances of embeddings, Relation Networks [48] leveraged relational module to represent an appropriate metric. TADAM [34] proposed metric scaling and metric task conditioning to boost the performance of Prototypical Networks.

Optimization-based meta-learning. Deep learning models are neither designed to train with very few examples nor to converge very fast. To fix that, optimization-based methods intend to learn with a few examples. Meta-learner [40] exploited an LSTM to satisfy two main desiderata of few-shot learning: quick acquisition of task-dependent knowledge and slow extraction of transferable knowledge. MAML [12] proposed a general optimization algorithm; it aims to find a set of model parameters, such that a small number of gradient steps with a small amount of training data from a new task will produce large improvements on that task. In that paper, first-order MAML was also proposed, which ignored the second-order derivatives of MAML. It achieved comparable results to complete MAML with orders of magnitude speedup. To further simplify MAML, Reptile [33] removed re-initialization for each task, making it a more natural choice in certain settings. LEO [44] proposed that it is beneficial to decouple the optimization-based meta-learning algorithms from high-dimensional model parameters. In particular, it learned a stochastic latent space from which the high-dimensional parameters can be generated. MetaOptNet [26] replaced the linear predictor with an SVM in the MAML framework; it incorporated a differentiable quadratic programming (QP) solver to allow end-to-end learning. For a complete list of recent works on meta-learning, we refer readers to [58].

Towards understanding MAML. To understand why MAML works in the first place, many efforts have been made either through an optimization perspective or a generalization perspective. Reptile [33] showed a variant of MAML works even without re-initialization for each task, because it tends to converge towards a solution that is close to each task's manifold of optimal solutions. In [39], the authors analyzed whether the effectiveness of MAML is due to rapid learning of each task or reusing the high quality features. It concluded that feature reuse is the dominant component in MAMLs efficacy, which is reaffirmed by experiments conducted in this paper.

Meta-learning datasets. Over the past several years, many datasets have been proposed to test meta-learning or few-shot learning algorithms. Omniglot [24] was one of the earliest few-shot learning datasets; it contains thousands of handwritten characters from the world's alphabets, intended for one-shot "visual Turing test". In [25], the authors reported the 3-year progress for the Omniglot challenge, concluding that human-level one-shot learnability is still hard for current meta-learning algorithms. [54] introduced mini-ImageNet, which is a subset of ImageNet [8]. In [42], a large portion of ImageNet was used for few-shot learning tests. Meta-dataset [53] summarized recent datasets and tested several representative methods in a uniform fashion.

Knowledge distillation. The idea of knowledge distillation (KD) dates back to [4]. The original idea was to compress the knowledge contained in an ensemble of models into a single smaller model. In [18], the authors generalized this idea and brought it into the deep learning framework.

In KD, knowledge is transferred from the teacher model to the student model by minimizing a loss in which the target is the distribution of class probabilities induced by the teacher model. In was shown in [61] that KD has several benefits for optimization and knowledge transfer between tasks. BAN [13] introduced sequential distillation, which also improved the performance of teacher models. In natural language processing (NLP), BAM [7] used BAN to distill from single-task models to a multi-task model, helping the multi-task model surpass its single-task teachers. Another two related works are [30] which provides theoretical analysis of self-distillation and CRD [50] which shows distillation improves the transferability across datasets.

3. Method

We establish preliminaries about the meta-learning problem and related algorithms in §3.1; then we present our baseline in §3.2; finally, we introduce how knowledge distillation helps few-shot learning in §3.3. For ease of comparison to previous work, we use the same notation as [26].

3.1. Problem formulation

The collection of meta-training tasks is defined as $\mathcal{T} = \{(\mathcal{D}_i^{train}, \mathcal{D}_i^{test})\}_{i=1}^{I}$, termed as meta-training set. The tuple $(\mathcal{D}_i^{train}, \mathcal{D}_i^{test})$ describes a training and a testing dataset of a task, where each dataset contains a small number of examples. Training examples $\mathcal{D}^{train} = \{(\mathbf{x}_t, y_t)\}_{t=1}^{T}$ and testing examples $\mathcal{D}^{test} = \{(\mathbf{x}_q, y_q)\}_{q=1}^{Q}$ are sampled from the same distribution.

A base learner $\mathcal{A}$, which is given by $y_* = f_\theta(\mathbf{x}_*)$ ($*$ denotes t or q), is trained on $\mathcal{D}^{train}$ and used as a predictor on $\mathcal{D}^{test}$. Due to the high dimensionality of $\mathbf{x}_*$, the base learner $\mathcal{A}$ suffers high variance. So training examples and testing examples are mapped into a feature space by an embedding model $\mathbf{\Phi}_* = f_\phi(\mathbf{x}_*)$. Assume the embedding model is fixed during training the base learner on each task, then the objective of the base learner is

$$\begin{aligned}\theta &= \mathcal{A}(\mathcal{D}^{train}; \phi) \\ &= \arg\min_{\theta} \mathcal{L}^{base}(\mathcal{D}^{train}; \theta, \phi) + \mathcal{R}(\theta),\end{aligned} \tag{1}$$

where $\mathcal{L}$ is the loss function and $\mathcal{R}$ is the regularization term.

The objective of the meta-learning algorithms is to learn a good embedding model, so that the average test error of the base learner on a distribution of tasks is minimized. Formally,

$$\phi = \arg\min_{\phi} \mathbb{E}_{\mathcal{T}}[\mathcal{L}^{meta}(\mathcal{D}^{test}; \theta, \phi)], \tag{2}$$

where $\theta = \mathcal{A}(\mathcal{D}^{train}; \phi)$.

Once meta-training is finished, the performance of the model is evaluated on a set of held-out tasks $\mathcal{S} = \{(\mathcal{D}_j^{train}, \mathcal{D}_j^{test})\}_{j=1}^{J}$, called meta-testing set. The evaluation is done over the distribution of the test tasks:

$$\mathbb{E}_{\mathcal{S}}[\mathcal{L}^{meta}(\mathcal{D}^{test}; \theta, \phi), \text{where } \theta = \mathcal{A}(\mathcal{D}^{train}; \phi)]. \tag{3}$$

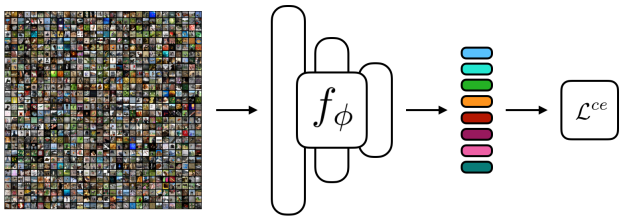

Figure 1. In meta-training, we train on an image classification task on the merged meta-training data to learn an embedding model. This model is then re-used at meta-testing time to extract embedding for a simple linear classifier.

3.2. Learning embedding model through classification

As we show in §3.1, the goal of meta-training is to learn a transferrable embedding model f_ϕ, which generalizes to any new task. Rather than designing new meta-learning algorithms to learn the embedding model, we propose that a model pre-trained on a classification task can generate powerful embeddings for the downstream base learner. To that end, we merge tasks from meta-training set into a single task, which is given by

$$\begin{aligned}\mathcal{D}^{new} &= \{(\mathbf{x}_i, y_i)\}_{k=1}^{K} \\ &= \cup\{\mathcal{D}_1^{train}, \ldots, \mathcal{D}_i^{train}, \ldots, \mathcal{D}_I^{train}\},\end{aligned} \tag{4}$$

where $\mathcal{D}_i^{train}$ is the task from $\mathcal{T}$. The embedding model is then

$$\phi = \arg\min_{\phi} \mathcal{L}^{ce}(\mathcal{D}^{new}; \phi), \tag{5}$$

and $\mathcal{L}^{ce}$ denotes the cross-entropy loss between predictions and ground-truth labels. We visualize the task in Figure 1.

As shown in Figure 2, for a task $(\mathcal{D}_j^{train}, \mathcal{D}_j^{test})$ sampled from meta-testing distribution, we train a base learner on $\mathcal{D}_j^{train}$. The base learner is instantiated as multivariate logistic regression. Its parameters $\theta = \{\boldsymbol{W}, \boldsymbol{b}\}$ include a weight term $\boldsymbol{W}$ and a bias term $\boldsymbol{b}$, given by

$$\theta = \arg\min_{\{\boldsymbol{W}, \boldsymbol{b}\}} \sum_{t=1}^{T} \mathcal{L}_t^{ce}(\boldsymbol{W} f_\phi(\mathbf{x}_t) + \boldsymbol{b}, y_t) + \mathcal{R}(\boldsymbol{W}, \boldsymbol{b}). \tag{6}$$

We also evaluate other base learners such as nearest neighbor classifier with $\mathcal{L}$-2 distance and/or cosine distance in §4.7.

In our method, the crucial difference between meta-training and meta-testing is the embedding model parameterized by ϕ is carried over from meta-training to meta-testing and kept unchanged when evaluated on tasks sampled from meta-testing set. The base learner is re-initialized

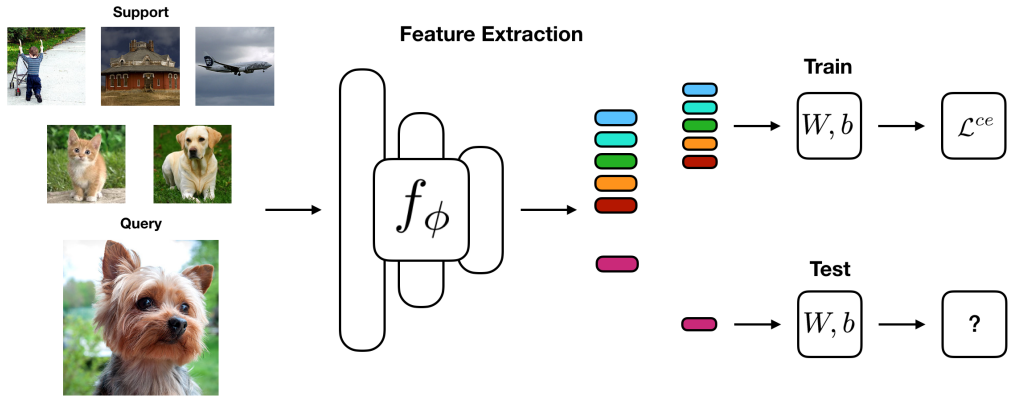

Figure 2. We show a meta-testing case for 5-way 1-shot task: 5 support images and 1 query image are transformed into embeddings using the fixed neural network; a linear model (logistic regression (LR) in this case) is trained on 5 support embeddings; the query image is tested using the linear model.

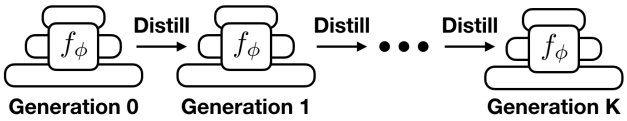

Figure 3. Sequential self-distillation: a vanilla model, termed as *Generation 0*, is trained with standard cross-entropy loss; then, the k-th generation is learned with knowledge distilled from the (k-1)-th generation.

for every task and trained on $\mathcal{D}^{train}$ of meta-testing task. Our method is the same with the pre-training phase of methods used in [44, 34]. Unlike other methods [9, 39], we *do not* fine-tune the embedding model f_ϕ during the meta-testing stage.

3.3. Sequential self-distillation

Knowledge distillation [18] is an approach to transfer knowledge embedded in an ensemble of models to a single model, or from a larger teacher model to a smaller student model. Instead of using the embedding model directly for meta-testing, we distill the knowledge from the embedding model to into a new model with an identical architecture, training on the same merged meta-training set. The new embedding model parameterized by ϕ' is trained to minimize a weighted sum of the cross-entropy loss between the predictions and ground-truth labels and the KullbackLeibler divergence (KL) between predictions and soft targets predicted by f_ϕ:

$$\phi' = \arg\min_{\phi'}(\alpha\mathcal{L}^{ce}(\mathcal{D}^{new};\phi')+ \beta KL(f(\mathcal{D}^{new};\phi'), f(\mathcal{D}^{new};\phi))), \tag{7}$$

where usually $\beta = 1-\alpha$.

We exploit the Born-again [13] strategy to apply KD sequentially to generate multiple generations, which is shown in Figure 3. At each step, the embedding model of k-th generation is trained with knowledge transferred from the embedding model of (k-1)-th generation:

$$\phi_k = \arg\min_{\phi}(\alpha\mathcal{L}^{ce}(\mathcal{D}^{new};\phi)+ \beta KL(f(\mathcal{D}^{new};\phi), f(\mathcal{D}^{new};\phi_{k-1}))). \tag{8}$$

Assume we repeat the operation K times, we use ϕ_K as the embedding model to extract features for meta-testing. We analyze the effects of sequential self-distillation in §4.6.

4. Experiments

We conduct experiments on four widely used few-shot image recognition benchmarks: miniImageNet [54], tieredImageNet [42], CIFAR-FS [3], and FC100 [34]. The first two are derivatives of ImageNet [43], while the last two are reorganized from the standard CIFAR-100 dataset [23, 51].

4.1. Setup

Architecture. Following previous works [29, 34, 26, 41, 9], we use a ResNet12 as our backbone: the network consists of 4 residual blocks, where each has 3 convolutional layers with 3×3 kernel; a 2×2 max-pooling layer is applied after each of the first 3 blocks; and a global average-pooling layer is on top of the fourth block to generate the feature embedding. Similar to [26], we use Dropblock as a regularizer and change the number of filters from (64,128,256,512) to (64,160,320,640). As a result, our ResNet12 is identical to that used in [41, 26] .

Optimization setup. We use SGD optimizer with a momentum of 0.9 and a weight decay of $5e^{-4}$. Each batch consists of 64 samples. The learning rate is initialized as 0.05 and decayed with a factor of 0.1 by three times for all datasets, except for miniImageNet where we only decay twice as the third decay has no effect. We train 100 epochs for miniImageNet, 60 epochs for tieredImageNet, and 90

model	backbone	miniImageNet 5-way		tieredImageNet 5-way	
		1-shot	5-shot	1-shot	5-shot
MAML [12]	32-32-32-32	48.70 ± 1.84	63.11 ± 0.92	51.67 ± 1.81	70.30 ± 1.75
Matching Networks [54]	64-64-64-64	43.56 ± 0.84	55.31 ± 0.73	-	-
IMP [2]	64-64-64-64	49.2 ± 0.7	64.7 ± 0.7	-	-
Prototypical Networks† [46]	64-64-64-64	49.42 ± 0.78	68.20 ± 0.66	53.31 ± 0.89	72.69 ± 0.74
TAML [21]	64-64-64-64	51.77 ± 1.86	66.05 ± 0.85	-	-
SAML [15]	64-64-64-64	52.22 ± n/a	66.49 ± n/a	-	-
GCR [27]	64-64-64-64	53.21 ± 0.80	72.34 ± 0.64	-	-
KTN(Visual) [35]	64-64-64-64	54.61 ± 0.80	71.21 ± 0.66	-	-
PARN[59]	64-64-64-64	55.22 ± 0.84	71.55 ± 0.66	-	-
Dynamic Few-shot [14]	64-64-128-128	56.20 ± 0.86	73.00 ± 0.64	-	-
Relation Networks [48]	64-96-128-256	50.44 ± 0.82	65.32 ± 0.70	54.48 ± 0.93	71.32 ± 0.78
R2D2 [3]	96-192-384-512	51.2 ± 0.6	68.8 ± 0.1	-	-
SNAIL [29]	ResNet-12	55.71 ± 0.99	68.88 ± 0.92	-	-
AdaResNet [32]	ResNet-12	56.88 ± 0.62	71.94 ± 0.57	-	-
TADAM [34]	ResNet-12	58.50 ± 0.30	76.70 ± 0.30	-	-
Shot-Free [41]	ResNet-12	59.04 ± n/a	77.64 ± n/a	63.52 ± n/a	82.59 ± n/a
TEWAM [37]	ResNet-12	60.07 ± n/a	75.90 ± n/a	-	-
MTL [47]	ResNet-12	61.20 ± 1.80	75.50 ± 0.80	-	-
Variational FSL [62]	ResNet-12	61.23 ± 0.26	77.69 ± 0.17	-	-
MetaOptNet [26]	ResNet-12	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
Diversity w/ Cooperation [11]	ResNet-18	59.48 ± 0.65	75.62 ± 0.48	-	-
Fine-tuning [9]	WRN-28-10	57.73 ± 0.62	78.17 ± 0.49	66.58 ± 0.70	85.55 ± 0.48
LEO-trainval† [44]	WRN-28-10	61.76 ± 0.08	77.59 ± 0.12	66.33 ± 0.05	81.44 ± 0.09
Ours-simple	ResNet-12	62.02 ± 0.63	79.64 ± 0.44	69.74 ± 0.72	84.41 ± 0.55
Ours-distill	ResNet-12	**64.82 ± 0.60**	**82.14 ± 0.43**	**71.52 ± 0.69**	**86.03 ± 0.49**

Table 1. **Comparison to prior work on miniImageNet and tieredImageNet.** Average few-shot classification accuracies (%) with 95% confidence intervals on miniImageNet and tieredImageNet meta-test splits. Results reported with input image size of 84x84. a-b-c-d denotes a 4-layer convolutional network with a, b, c, and d filters in each layer. † results obtained by training on the union of training and validation sets.

epochs for both CIFAR-FS and FC100. During distillation, we use the same learning schedule and set $\alpha = \beta = 0.5$.

Data augmentation. When training the embedding network on transformed meta-training set, we adopt random crop, color jittering, and random horizontal flip as in [26]. For meta-testing stage, we train an N-way logistic regression base classifier. We use the implementations in scikit-learn [1] for the base classifier.

4.2. Results on ImageNet derivatives

The *miniImageNet* dataset [54] is a standard benchmark for few-shot learning algorithms for recent works. It consists of 100 classes randomly sampled from the ImageNet; each class contains 600 downsampled images of size 84x84. We follow the widely-used splitting protocol proposed in [40], which uses 64 classes for meta-training, 16 classes for meta-validation, and the remaining 20 classes for meta-testing.

The *tieredImageNet* dataset [42] is another subset of ImageNet but has more classes (608 classes). These classes are first grouped into 34 higher-level categories, which are further divided into 20 training categories (351 classes), 6 validation categories (97 classes), and 8 testing categories (160 classes). Such construction ensures the training set is distinctive enough from the testing set and makes the problem more challenging.

Results. During meta-testing, we evaluate our method with 3 runs, where in each run the accuracy is the mean accuracy of 1000 randomly sampled tasks. We report the median of 3 runs in Table 1. Our simple baseline with ResNet-12 is already comparable with the state-of-the-art MetaOptNet [26] on miniImageNet, and outperforms all previous works by at least 3% on tieredImageNet. The network trained with distillation further improves over the simple baseline by 2-3%.

We notice that previous works [38, 44, 34, 47] have also leveraged the standard cross-entropy pre-training on the meta-training set. In [34, 44], a wide ResNet (WRN-28-10) is trained to classify all classes in the meta-training set (or combined meta-training and meta-validation set), and then frozen during the meta-training stage. [9] also con-

model	backbone	CIFAR-FS 5-way		FC100 5-way	
		1-shot	5-shot	1-shot	5-shot
MAML [12]	32-32-32-32	58.9 ± 1.9	71.5 ± 1.0	-	-
Prototypical Networks [46]	64-64-64-64	55.5 ± 0.7	72.0 ± 0.6	35.3 ± 0.6	48.6 ± 0.6
Relation Networks [48]	64-96-128-256	55.0 ± 1.0	69.3 ± 0.8	-	-
R2D2 [3]	96-192-384-512	65.3 ± 0.2	79.4 ± 0.1	-	-
TADAM [34]	ResNet-12	-	-	40.1 ± 0.4	56.1 ± 0.4
Shot-Free [41]	ResNet-12	69.2 ± n/a	84.7 ± n/a	-	-
TEWAM [37]	ResNet-12	70.4 ± n/a	81.3 ± n/a	-	-
Prototypical Networks [46]	ResNet-12	72.2 ± 0.7	83.5 ± 0.5	37.5 ± 0.6	52.5 ± 0.6
MetaOptNet [26]	ResNet-12	72.6 ± 0.7	84.3 ± 0.5	41.1 ± 0.6	55.5 ± 0.6
Ours-simple	ResNet-12	71.5 ± 0.8	86.0 ± 0.5	42.6 ± 0.7	59.1 ± 0.6
Ours-distill	ResNet-12	**73.9 ± 0.8**	**86.9 ± 0.5**	**44.6 ± 0.7**	**60.9 ± 0.6**

Table 2. **Comparison to prior work on CIFAR-FS and FC100.** Average few-shot classification accuracies (%) with 95% confidence intervals on CIFAR-FS and FC100. a-b-c-d denotes a 4-layer convolutional network with a, b, c, and d filters in each layer.

ducts pre-training but the model is fine-tuned using the support images in meta-testing set, achieving 57.73 ± 0.62. We adopt the same architecture and gets 61.1 ± 0.86. So fine-tuning on small set of samples makes the performance worse. Another work [34] adopts a multi-task setting by jointly training on the standard classification task and few-shot classification (5-way) task. In another work [47], the ResNet-12 is pre-trained before mining hard tasks for the meta-training stage. In this work, we show standard cross-entropy pre-training is sufficient to generate strong embeddings without meta-learning techniques or any fine-tuning.

4.3. Results on CIFAR derivatives

The *CIFAR-FS* dataset [3] is a derivative of the original CIFAR-100 dataset by randomly splitting 100 classes into 64, 16 and 20 classes for training, validation, and testing, respectively. The *FC100* dataset [34] is also derived from CIFAR-100 dataset in a similar way to tieredImagNet. This results in 60 classes for training, 20 classes for validation, and 20 classes for testing.

Results. Similar to previous experiments, we evaluate our method with 3 runs, where in each run the accuracy is the mean accuracy of 3000 randomly sampled tasks. Table 2 summarizes the results, which shows that our simple baseline is comparable to Prototypical Networks [46] and MetaOptNet [26] on CIFAR-FS dataset, and outperforms both of them on FC100 dataset. Our distillation version achieves the new state-of-the-art on both datasets. This verifies our hypothesis that a good embedding plays an important role in few-shot recognition.

4.4. Embeddings from self-supervised representation learning

Using unsupervised learning to improve the generalization of the meta-learning algorithms [57] removes the needs of data annotation. In addition to using embeddings from supervised pre-training, we also train a linear classifier on embeddings from self-supervised representation learning. Following similar training protocols in Moco [16] and CMC [49], we train a ResNet50 [17] (without using labels) on the merged meta-training data to learn an embedding model. We compare unsupervised ResNet50 to a supervised ResNet50. From Table 3, we observe that using embeddings from self-supervised ResNet50 is only slightly worse than using embeddings from supervised ResNet50 (in 5-shot setting, the results are comparable). This observation shows the potential of self-supervised learning in the scenario of few-shot learning.

model	backbone	miniImageNet 5-way	
		1-shot	5-shot
Supervised	ResNet50	**57.56 ± 0.79**	73.81 ± 0.63
Moco [16]	ResNet50	54.19 ± 0.93	73.04 ± 0.61
CMC [49]	ResNet50*	56.10 ± 0.89	**73.87 ± 0.65**

Table 3. Comparsions of embeddings from supervised pre-training and self-supervised pre-training (Moco and CMC). * the encoder of each view is 0.5× width of a normal ResNet-50.

4.5. Ablation experiments

In this section, we conduct ablation studies to analyze how each component affects the few-shot recognition performance. We study the following five components of our method: (a) we chose logistic regression as our base learner, and compare it to a nearest neighbour classifier with euclidean distance; (b) we find that normalizing the feature vectors onto the unit sphere, e.g., $\mathcal{L}$-2 normalization, could improve the classification of the downstream base classi-

NN	LR	$\mathcal{L}$-2	Aug	Distill	miniImageNet 1-shot	miniImageNet 5-shot	tieredImageNet 1-shot	tieredImageNet 5-shot	CIFAR-FS 1-shot	CIFAR-FS 5-shot	FC100 1-shot	FC100 5-shot
✓					56.29	69.96	64.80	78.75	64.36	78.00	38.40	49.12
	✓				58.74	78.31	67.62	84.77	66.92	84.78	40.36	57.23
	✓	✓			61.56	79.27	69.53	85.08	71.24	85.63	42.77	58.86
	✓	✓	✓		62.02	79.64	69.74	85.23	71.45	85.95	42.59	59.13
	✓	✓	✓	✓	64.82	82.14	71.52	86.03	73.89	86.93	44.57	60.91

Table 4. **Ablation study on four benchmarks with ResNet-12 as backbone network.** "NN" and "LR" stand for nearest neighbour classifier and logistic regression. "$\mathcal{L}$-2" means feature normalization after which feature embeddings are on the unit sphere. "Aug" indicates that each support image is augmented into 5 samples to train the classifier. "Distill" represents the use of knowledge distillation.

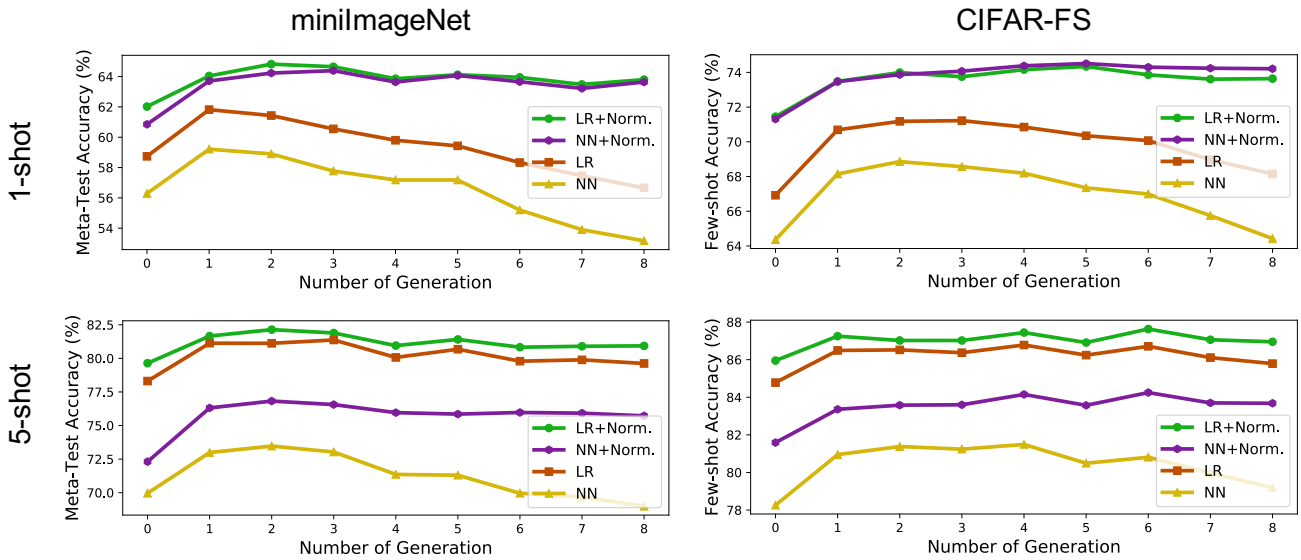

Figure 4. **Evaluation on different generations of distilled networks.** The 0-th generation (or root generation) indicates the vanilla network trained with only standard classification cross-entropy loss. The k-th generation is trained by combining the standard classification loss and the knowledge distillation (KD) loss using the (k-1)-th generation as the teacher model. Logistic regression (LR) and nearest neighbours (NN) are evaluated.

fier; (c) during meta-testing, we create 5 augmented samples from each support image to alleviate the data insufficiency problem, and using these augmented samples to train the linear classifier; (d) we distill the embedding network on the training set by following the sequential distillation [13] strategy.

Table 4 shows the results of our ablation studies on miniImageNet, tieredImageNet, CIFAR-FS, and FC100. In general, logistic regression significantly outperforms the nearest neighbour classifier, especially for the 5-shot case; $\mathcal{L}$-2 normalization consistently improves the 1-shot accuracy by 2% on all datasets; augmenting the support images leads to marginal improvement; even with all these techniques, distillation can still provide 2% extra gain.

4.6. Effects of distillation

We can use sequential self-distillation to get an embedding model, similar to the one in Born-again networks [13]. We therefore investigate the effect of this strategy on the performance of downstream few-shot classification.

In addition to logistic regression and nearest-neighbour classifiers, we also look into a cosine similarity classifier, which is equivalent to the nearest-neighbour classifier but with normalized features (noted as "NN+Norm."). The plots of 1-shot and 5-shot results on miniImageNet and CIFAR-FS are shown in Figure 4. The 0-th generation (or root generation) refers to the vanilla model trained with only standard cross-entropy loss, and the (k-1)-th generation is distilled into k-th generation. In general, few-shot recognition performance keeps getting better in the first two or three generations. After certain number of generations, the accuracy starts decreasing for logistic regression and nearest neighbour. Normalizing the features can significantly alleviate this problem.

In Table 1, Table 2, and Table 4, we evalute the model of the second generation on miniImageNet, CIFAR-FS and

model	backbone	miniImageNet 5-way		tieredImageNet 5-way	
		1-shot	5-shot	1-shot	5-shot
Ours	64-64-64-64	55.25 ± 0.58	71.56 ± 0.52	56.18 ± 0.70	72.99 ± 0.55
Ours-distill	64-64-64-64	55.88 ± 0.59	71.65 ± 0.51	56.76 ± 0.68	73.21 ± 0.54
Ours-trainval	64-64-64-64	56.32 ± 0.58	72.46 ± 0.52	56.53 ± 0.68	73.15 ± 0.58
Ours-distill-trainval	64-64-64-64	$\mathbf{56.64 \pm 0.58}$	$\mathbf{72.85 \pm 0.50}$	$\mathbf{57.35 \pm 0.70}$	$\mathbf{73.98 \pm 0.56}$
Ours	ResNet-12	62.02 ± 0.63	79.64 ± 0.44	69.74 ± 0.72	84.41 ± 0.55
Ours-distill	ResNet-12	64.82 ± 0.60	82.14 ± 0.43	71.52 ± 0.69	86.03 ± 0.49
Ours-trainval	ResNet-12	63.59 ± 0.61	80.86 ± 0.47	71.12 ± 0.68	85.94 ± 0.46
Ours-distill-trainval	ResNet-12	$\mathbf{66.58 \pm 0.65}$	$\mathbf{83.22 \pm 0.39}$	$\mathbf{72.98 \pm 0.71}$	$\mathbf{87.46 \pm 0.44}$
Ours	SEResNet-12	62.29 ± 0.60	79.94 ± 0.46	70.31 ± 0.70	85.22 ± 0.50
Ours-distill	SEResNet-12	65.96 ± 0.63	82.05 ± 0.46	71.72 ± 0.69	86.54 ± 0.49
Ours-trainval	SEResNet-12	64.07 ± 0.61	80.92 ± 0.43	71.76 ± 0.66	86.27 ± 0.45
Ours-distill-trainval	SEResNet-12	$\mathbf{67.73 \pm 0.63}$	$\mathbf{83.35 \pm 0.41}$	$\mathbf{72.55 \pm 0.69}$	$\mathbf{86.72 \pm 0.49}$

Table 5. Comparisons of different backbones on *miniImageNet* and *tieredImageNet*.

FC100 datasets; we use the first generation on tieredImageNet. Model selection is done on the validation set.

4.7. Choice of base classifier

One might argue in the 1-shot case, that a linear classifier should behavior similarly to a nearest-neighbour classifier. However in Table 4 and Figure 4, we find that logistic regression is clearly better than nearest-neighbour. We argue that this is casued by the scale of the features. After we normalize the features by the $\mathcal{L}$-2 norm, logistic regression ("LR+Norm") performs similarly to the nearest neighbour classifier ("NN+Norm."), as shown in the first row of Figure 4. However, when increasing the size of the support set to 5, logistic regression is significantly better than nearest-neighbour even after feature normalization

4.8. Multi-task vs multi-way classification?

We are interested in understanding whether the efficacy of our simple baseline is due to multi-task or multi-way classification. We compare to training an embedding model through *multi-task* learning: a model with shared embedding network and different classification heads is constructed, where each head is only classifying the corresponding category; then we use the embedding model to extract features as we do with our baseline model. This achieves 58.53 ± 0.8 on mini-ImageNet 5-way 1-shot case, compared to our baseline model which is 62.02 ± 0.63. So we argue that the speciality of our setting, where the few-shot classification tasks are mutually exclusive and can be merged together into a single *multi-way* classification task, makes the simple model effective.

4.9. Comparsions of different network backbones.

Better backbone networks generally produce better results; this is also obvious in few-shot learning and/or meta-learning (as shown in Table 1). To further verify our assumption that the key success of few-shot learning algorithms is due to the quality of embeddings, we compare three alternatives in Table 5 and Table 6: a ConvNet with four four convolutional layers (64, 64, 64, 64); a ResNet12 as in Table 1; a ResNet12 with sequeeze-and-excitation [19] modules. For each model, we have four settings: training on meta-training set; training and distilling on meta-training set; training on meta-training set and meta-validation set; training and distilling on meta-training set and meta-validation set. The results consistently improve with more data and better networks. This is inline with our hypothesis: embeddings are the most critical factor to the performance of few-shot learning/meta learning algorithms; better embeddings will lead to better few-shot testing performance (even with a simple linear classier). In addition, our ConvNet model also outperforms other few-shot learning and/or meta learning models using the same network. This verifies that in both small model regime (ConvNet) and large model regime (ResNet), few-shot learning and meta learning algorithms are *no better* than learning a good embedding model.

5. Discussion

We have proposed a simple baseline for few-shot image classification in the meta-learning context. This approach has been underappreciated in the literature thus far. We show with numerous experiments that uch a simple baseline

model	backbone	CIFAR-FS 5-way		FC100 5-way	
		1-shot	5-shot	1-shot	5-shot
Ours	64-64-64-64	62.7 ± 0.8	78.7 ± 0.5	39.6 ± 0.6	53.5 ± 0.5
Ours-distill	64-64-64-64	63.8 ± 0.8	79.5 ± 0.5	40.3 ± 0.6	54.1 ± 0.5
Ours-trainval	64-64-64-64	63.5 ± 0.8	79.8 ± 0.5	43.2 ± 0.6	58.5 ± 0.5
Ours-distill-trainval	64-64-64-64	$\mathbf{64.9 \pm 0.8}$	$\mathbf{80.3 \pm 0.5}$	$\mathbf{44.6 \pm 0.6}$	$\mathbf{59.2 \pm 0.5}$
Ours	ResNet-12	71.5 ± 0.8	86.0 ± 0.5	42.6 ± 0.7	59.1 ± 0.6
Ours-distill	ResNet-12	73.9 ± 0.8	86.9 ± 0.5	44.6 ± 0.7	60.9 ± 0.6
Ours-trainval	ResNet-12	73.1 ± 0.8	86.7 ± 0.5	49.5 ± 0.7	66.4 ± 0.6
Ours-distill-trainval	ResNet-12	$\mathbf{75.4 \pm 0.8}$	$\mathbf{88.2 \pm 0.5}$	$\mathbf{51.6 \pm 0.7}$	$\mathbf{68.4 \pm 0.6}$
Ours	SEResNet-12	72.0 ± 0.8	86.0 ± 0.6	43.4 ± 0.6	59.1 ± 0.6
Ours-distill	SEResNet-12	74.2 ± 0.8	87.2 ± 0.5	44.9 ± 0.6	61.4 ± 0.6
Ours-trainval	SEResNet-12	73.3 ± 0.8	86.8 ± 0.5	49.9 ± 0.7	66.8 ± 0.6
Ours-distill-trainval	SEResNet-12	$\mathbf{75.6 \pm 0.8}$	$\mathbf{88.2 \pm 0.5}$	$\mathbf{52.0 \pm 0.7}$	$\mathbf{68.8 \pm 0.6}$

Table 6. Comparisons of different backbones on *CIFAR-FS* and *FC100*.

outperforms the current state-of-the-arts on four widely-used few-shot benchmarks. Combined with self-distillation, the performance further improves by 2-3%. Even when meta-training labels are unavailable, it may be possible to leverage state of the art self-supervised learning approaches to learn very good embeddings for meta-testing tasks.

1. What is the intuition of this paper?
A: We hope this paper will shed new light on few-shot classification. We believe representations play an important role. Shown by our empirical experiments, a linear model can generalize well as long as a good representation of the data is given.

2. Why does this simple baseline work? Is there anything that makes few-shot classification special?
A: Few-shot classification is a special case of meta-learning in terms of compositionality of tasks. Each task is an K-way classification problem, and on current benchmarks the classes, even between tasks, are all mutually exclusive. This means we can merge all N of the K-way classification tasks into a single but harder NK-way classification task. Our finding is that training an embedding model on this new NK-way task turns out to transfer well to meta-testing set. On the other hand, we also find that self-supervised embedding, which does not explicitly require this NK compositionality, achieves a similar level of performance. A concurrent work [10] studies the representations for few-shot learning from the theoretical point of view.

3. Does your work negate recent progress in meta-learning?
A: No. Meta-learning is much broader than just few-shot classification. Although we show a simple baseline outperforms other complicated meta-learning algorithms in few-shot classification, methods like MAML may still be favorable in other meta-learning domains (e.g., meta-reinforcement learning).

4. Why does distillation work? What does it suggest?
A: The soft-labels [18] from the teacher model depict the fact that some classes are closer to each other than other classes. For example, a white goat is much more similar to a brown horse than to an airplane. But the one-hot label does not capture this. After being regularized by soft-labels, the network learns to capture the metric distance. From theoretical perspective, [36] provides analysis for linear case. Ongoing work [31] argues distillation amplifies regularization in Hilbert space.

References

[1] Machine learning in python. https://scikit-learn.org/stable/. 5

[2] Kelsey Allen, Evan Shelhamer, Hanul Shin, and Joshua Tenenbaum. Infinite mixture prototypes for few-shot learning. In *ICML*, 2019. 5

[3] Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018. 4, 5, 6

[4] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, 2006. 2

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *ICLR*, 2019. 1

[6] Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning. *ArXiv*, abs/2003.04390, 2020. 1

[7] Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. Bam! born-again multi-task networks for natural language understanding. In *ACL*, 2019. 3

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 2

[9] Guneet Singh Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *ICLR*, 2020. 1, 4, 5, 6

[10] Simon Shaolei Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. *ArXiv*, abs/2002.09434, 2020. 9

[11] Nikita Dvornik, Cordelia Schmid, and Julien Mairal. Diversity with cooperation: Ensemble methods for few-shot classification. In *ICCV*, 2019. 5

[12] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 1, 2, 5, 6

[13] Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born-again neural networks. In *ICML*, 2018. 3, 4, 7

[14] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018. 5

[15] Fusheng Hao, Fengxiang He, Jun Cheng, Lei Wang, Jianzhong Cao, and Dacheng Tao. Collect and select: Semantic alignment metric learning for few-shot learning. In *ICCV*, 2019. 5

[16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *ArXiv*, abs/1911.05722, 2019. 6, 12

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016. 6

[18] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. 2, 4, 9

[19] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 8

[20] Shaoli Huang and Dacheng Tao. All you need is a good representation: A multi-level and classifier-centric representation for few-shot learning. *ArXiv*, abs/1911.12476, 2019. 1

[21] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *CVPR*, 2019. 5

[22] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015. 2

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 4

[24] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 2015. 2

[25] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. The omniglot challenge: a 3-year progress report. *Current Opinion in Behavioral Sciences*, 2019. 2

[26] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019. 1, 2, 3, 4, 5, 6

[27] Aoxue Li, Tiange Luo, Tao Xiang, Weiran Huang, and Liwei Wang. Few-shot learning with global class representations. In *ICCV*, 2019. 5

[28] Hongyang Li, David Eigen, Samuel Dodge, Matthew Zeiler, and Xiaogang Wang. Finding task-relevant features for few-shot learning by category traversal. In *CVPR*, 2019. 1

[29] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017. 4, 5

[30] Hossein Mobahi, Mehrdad Farajtabar, and Peter L Bartlett. Self-distillation amplifies regularization in hilbert space. *arXiv preprint arXiv:2002.05715*, 2020. 3

[31] Hossein Mobahi, Mehrdad Farajtabar, and Peter L. Bartlett. Self-distillation amplifies regularization in hilbert space. *ArXiv*, abs/2002.05715, 2020. 9

[32] Tsendsuren Munkhdalai, Xingdi Yuan, Soroush Mehri, and Adam Trischler. Rapid adaptation with conditionally shifted neurons. *arXiv preprint arXiv:1712.09926*, 2017. 5

[33] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *ArXiv*, abs/1803.02999, 2018. 2

[34] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NIPS*, 2018. 1, 2, 4, 5, 6

[35] Zhimao Peng, Zechao Li, Junge Zhang, Yan Li, Guo-Jun Qi, and Jinhui Tang. Few-shot image recognition with knowledge transfer. In *ICCV*, 2019. 5

[36] Mary Phuong and Christoph Lampert. Towards understanding knowledge distillation. In *ICML*, 2019. 9

[37] 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. 5, 6

[38] Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L. Yuille. Few-shot image recognition by predicting parameters from activations. In *CVPR*, 2018. 5

[39] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*, 2019. 1, 2, 4

[40] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 2, 5

[41] Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Few-shot learning with embedded class models and shot-free meta training. In *ICCV*, 2019. 4, 5, 6

[42] Mengye Ren, Sachin Ravi, Eleni Triantafillou, Jake Snell, Kevin Swersky, Josh B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 2, 4, 5

[43] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 4

[44] 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. 1, 2, 4, 5

[45] Tyler Scott, Karl Ridgeway, and Michael C Mozer. Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning. In *NIPS*, 2018. 1

[46] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017. 1, 2, 5, 6

[47] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *CVPR*, 2019. 5, 6

[48] 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, 5, 6

[49] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 6, 12

[50] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*, 2019. 3

[51] Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *TPAMI*, 2008. 4

[52] Eleni Triantafillou, Richard S. Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens. In *NIPS*, 2017. 1

[53] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. *ArXiv*, 2019. 2

[54] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NIPS*, 2016. 1, 2, 4, 5

[55] Yuxiong Wang and Martial Hebert. Learning to learn: Model regression networks for easy small sample learning. In *ECCV*, 2016. 1

[56] Yu-Xiong Wang, Ross B. Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. *CVPR*, 2018. 1

[57] Yu-Xiong Wang and Martial Hebert. Learning from small sample sets by combining unsupervised meta-training with cnns. In *Advances in Neural Information Processing Systems 29*, 2016. 6

[58] Lilian Weng. Meta-learning: Learning to learn fast. *lilianweng.github.io/lil-log*, 2018. 2

[59] Ziyang Wu, Yuwei Li, Lihua Guo, and Kui Jia. Parn: Position-aware relation networks for few-shot learning. In *ICCV*, 2019. 5

[60] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Learning embedding adaptation for few-shot learning. *CoRR*, abs/1812.03664, 2018. 1

[61] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 2017. 3

[62] Jian Zhang, Chenglong Zhao, Bingbing Ni, Minghao Xu, and Xiaokang Yang. Variational few-shot learning. In *ICCV*, 2019. 5

A. Architectures

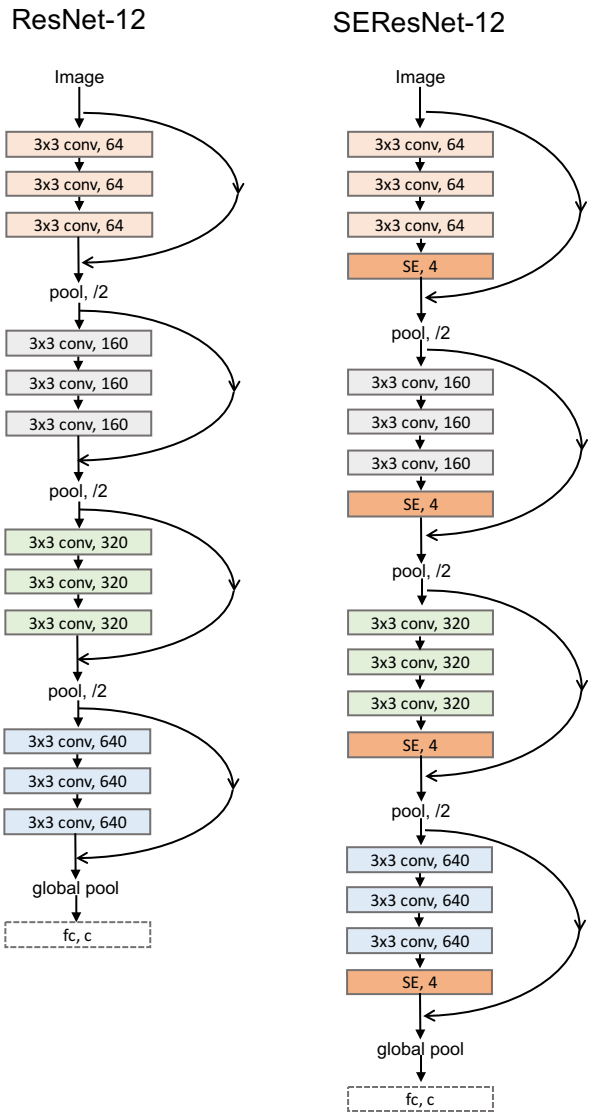

Figure 5. Nework architectures of ResNet-12 and SEResNet-12 used in this paper. The "SE, 4" stands for a Squeeze-and-Excitation layer with reduction parameter of 4. Dotted box will be removed during meta-testing stage.

The architectures of ResNet-12 ans SEResNet-12 is show in Figure 5.

B. More Training Details

For SEResNet-12, we use the same training setup as ResNet-12 on all four benchmarks, as described in Sec 4.1.

For 4-layer convnet, we also the same training setup as ResNet-12 on tieredImageNet, CIFAR-FS, and FC100, For miniImageNet, we train for 240 epochs with learning rate decayed at epochs 150, 180, and 210 with a factor of 0.1. We found that using the logit layer as feature results in slightly better accuracy ($\leq 1\%$) on miniImageNet, so we report this number in Table 5 for miniImageNet.

C. Unsupervised Learning Details

We adapt the first layer of a standard ResNet-50 to take images of size 84×84 as input. We only train on the meta-train set of miniImageNet dataset (do not use meta-val set). We follow the training recipe in CMC [49] and MoCo [16] except for two differences. The first one is that we only use 2048 negatives for each positive sample as miniImageNet contains less than 40k images in total. The second difference is that we train for 2000 epochs, with a learning rate initialized as 0.03 and decayed by consine annealing.