

# Few-Shot Image Recognition by Predicting Parameters from Activations

Siyuan Qiao<sup>1</sup> Chenxi Liu<sup>1</sup> Wei Shen<sup>1,2</sup> Alan Yuille<sup>1</sup>  
 Johns Hopkins University<sup>1</sup> Shanghai University<sup>2</sup>  
 {siyuan.qiao, cxliu, alan.yuille}@jhu.edu wei.shen@t.shu.edu.cn

## Abstract

In this paper, we are interested in the few-shot learning problem. In particular, we focus on a challenging scenario where the number of categories is large and the number of examples per novel category is very limited, e.g. 1, 2, or 3. Motivated by the close relationship between the parameters and the activations in a neural network associated with the same category, we propose a novel method that can adapt a pre-trained neural network to novel categories by directly predicting the parameters from the activations. Zero training is required in adaptation to novel categories, and fast inference is realized by a single forward pass. We evaluate our method by doing few-shot image recognition on the ImageNet dataset, which achieves the state-of-the-art classification accuracy on novel categories by a significant margin while keeping comparable performance on the large-scale categories. We also test our method on the MiniImageNet dataset and it strongly outperforms the previous state-of-the-art methods.

## 1. Introduction

Recent years have witnessed rapid advances in deep learning [20], with a particular example being visual recognition [11, 16, 28] on large-scale image datasets, e.g., ImageNet [27]. Despite their great performances on benchmark datasets, the machines exhibit clear difference with people in the way they learn concepts. Deep learning methods typically require huge amounts of supervised training data per concept, and the learning process could take days using specialized hardware, i.e. GPUs. In contrast, children are known to be able to learn novel visual concepts almost effortlessly with a few examples after they have accumulated enough past knowledge [2]. This phenomenon motivates computer vision research on the problem of few-shot learning, i.e., the task to learn novel concepts from only a few examples for each category [7, 18].

Formally, in the few-shot learning problem [14, 24, 29], we are provided with a large-scale set  $\mathcal{D}_{\text{large}}$  with categories  $\mathcal{C}_{\text{large}}$  and a few-shot set  $\mathcal{D}_{\text{few}}$  with categories  $\mathcal{C}_{\text{few}}$  that do not overlap with  $\mathcal{C}_{\text{large}}$ .  $\mathcal{D}_{\text{large}}$  has sufficient training samples for

each category whereas  $\mathcal{D}_{\text{few}}$  has only a few examples ( $< 6$  in this paper). The goal is to achieve good classification performances, either on  $\mathcal{D}_{\text{few}}$  or on both  $\mathcal{D}_{\text{few}}$  and  $\mathcal{D}_{\text{large}}$ . We argue that a good classifier should have the following properties: (1) It achieves reasonable performance on  $\mathcal{C}_{\text{few}}$ . (2) Adapting to  $\mathcal{C}_{\text{few}}$  does not degrade the performance on  $\mathcal{C}_{\text{large}}$  significantly (if any). (3) It is fast in inference and adapts to few-shot categories with little or zero training, i.e., an efficient lifelong learning system [3, 4].

Both parametric and non-parametric methods have been proposed for the few-shot learning problem. However, due to the limited number of samples in  $\mathcal{D}_{\text{few}}$  and the imbalance between  $\mathcal{D}_{\text{large}}$  and  $\mathcal{D}_{\text{few}}$ , parametric models usually fail to learn well from the training samples [24]. On the other hand, many non-parametric approaches such as nearest neighbors can adapt to the novel concepts easily without severely forgetting the original classes. But this requires careful designs of the distance metrics [1], which can be difficult and sometimes empirical. To remedy this, some previous work instead adapts feature representation to the metrics by using siamese networks [14, 22]. As we will show later through experiments, these methods do not fully satisfy the properties mentioned above.

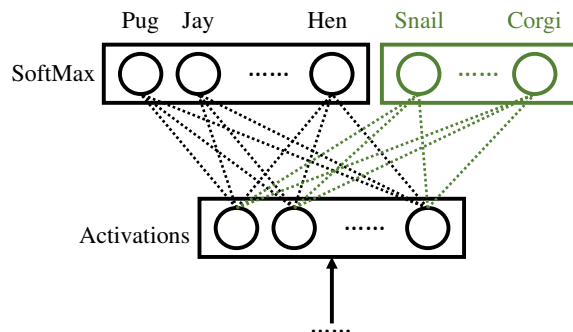


Figure 1: Illustration of pre-training on  $\mathcal{D}_{\text{large}}$  (black) and few-shot novel category adaptation to  $\mathcal{D}_{\text{few}}$  (green). The green circles are the novel categories, and the green lines represent the unknown parameters for categories in  $\mathcal{C}_{\text{few}}$ .

In this paper, we present an approach that meets the desired properties well. Our method starts with a pre-trained deep neural network on  $\mathcal{D}_{\text{large}}$ . The final classification layers

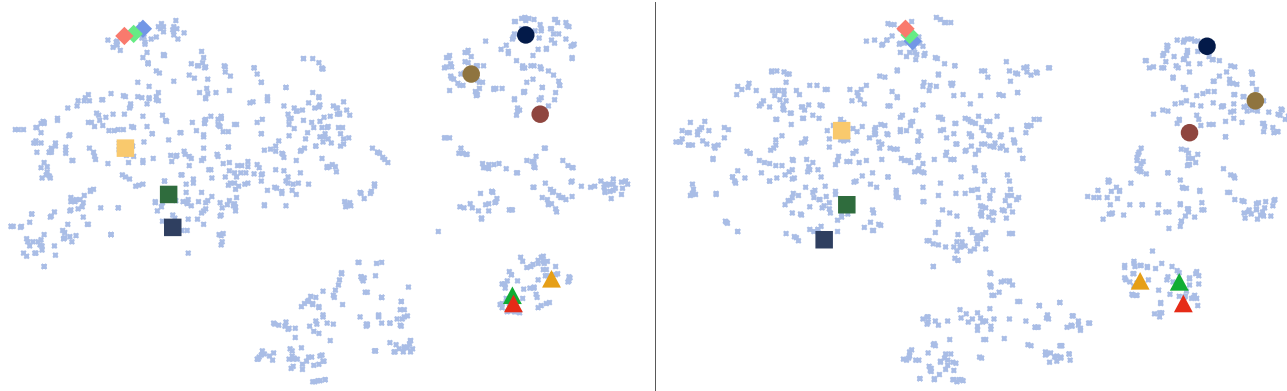


Figure 2: Our motivation: t-SNE [23] results on the average activations  $\bar{\mathbf{a}}_y$  of each category before the fully connected layer of a 50-layer ResNet [11] pre-trained on  $\mathcal{D}_{\text{large}}$  from ImageNet [27] (left) and the parameters  $\mathbf{w}_y$  of each category in the last fully connected layer (right). Each point represents a category. Highlighted points with the same color and shape correspond to the same category. Circles are *mammals*, triangles are *birds*, diamonds are *buses*, and squares are *home appliances*.

(the fully connected layer and the softmax layer) are shown in Figure 1. We use  $\mathbf{w}_y \in \mathbb{R}^n$  to denote the parameters for category  $y$  in the fully connected layer, and use  $\mathbf{a}(x) \in \mathbb{R}^n$  to denote the activations before the fully connected layer of an image  $x$ . Training on  $\mathcal{D}_{\text{large}}$  is standard; the real challenge is how to re-parameterize the last fully connected layer to include the novel categories under the few-shot constraints, *i.e.*, for each category in  $\mathcal{C}_{\text{few}}$  we have only a few examples. Our proposed method addresses this challenge by directly predicting the parameters  $\mathbf{w}_y$  (in the fully connected layer) using the activations belonging to that category, *i.e.*  $\mathcal{A}_y = \{\mathbf{a}(x) | x \in \mathcal{D}_{\text{large}} \cup \mathcal{D}_{\text{few}}, Y(x) = y\}$ , where  $Y(\cdot)$  denotes the category of the image.

This parameter predictor stems from the tight relationship between the parameters and activations. Intuitively in the last fully connected layer, we want  $\mathbf{w}_y \cdot \mathbf{a}_y$  to be large, for all  $\mathbf{a}_y \in \mathcal{A}_y$ . Let  $\bar{\mathbf{a}}_y \in \mathbb{R}^n$  be the mean of the activations in  $\mathcal{A}_y$ . Since it is known that the activations of images in the same category are spatially clustered together [5], a reasonable choice of  $\mathbf{w}_y$  is to align with  $\bar{\mathbf{a}}_y$  in order to maximize the inner product, and this argument holds true for all  $y$ . To verify this intuition, we use t-SNE [23] to visualize the neighbor embeddings of the activation statistic  $\bar{\mathbf{a}}_y$  and the parameters  $\mathbf{w}_y$  for each category of a pre-trained deep neural network, as shown in Figure 2. Comparing them and we observe a high similarity in both the local and the global structures. More importantly, the semantic structures [13] are also preserved in both activations and parameters, indicating a promising generalizability to unseen categories.

These results suggest the existence of a category-agnostic mapping from the activations to the parameters given a good feature extractor  $\mathbf{a}(\cdot)$ . In our work, we parameterize this mapping with a feedforward network that is learned by back-propagation. This mapping, once learned,

is used to predict parameters for both  $\mathcal{C}_{\text{few}}$  and  $\mathcal{C}_{\text{large}}$ .

We evaluate our method on two datasets. The first one is MiniImageNet [29], a simplified subset of ImageNet ILSVRC 2015 [27], in which  $\mathcal{C}_{\text{large}}$  has 80 categories and  $\mathcal{C}_{\text{few}}$  has 20 categories. Each category has 600 images of size  $84 \times 84$ . This small dataset is the benchmark for natural images that the previous few-shot learning methods are evaluated on. However, this benchmark only reports the performances on  $\mathcal{D}_{\text{few}}$ , and the accuracy is evaluated under 5-way test, *i.e.*, to predict the correct category from only 5 category candidates. In this paper, we will take a step forward by evaluating our method on the full ILSVRC 2015 [27], which has 1000 categories. We split the categories into two sets where  $\mathcal{C}_{\text{large}}$  has 900 and  $\mathcal{C}_{\text{few}}$  has the rest 100. The methods will be evaluated under 1000-way test on both  $\mathcal{D}_{\text{large}}$  and  $\mathcal{D}_{\text{few}}$ . This is a setting that is considerably larger than what has been experimented in the few-shot learning before. We compare our method with the previous work and show state-of-the-art performances.

The rest of the paper is organized as follows: §2 defines and explains our model, §3 presents the related work, §4 shows the experimental results, and §5 concludes the paper.

## 2. Model

The key component of our approach is the category-agnostic parameter predictor  $\phi : \bar{\mathbf{a}}_y \rightarrow \mathbf{w}_y$  (Figure 3). More generally, we could allow the input to  $\phi$  to be a statistic representing the activations of category  $y$ . Note that we use the same mapping function for all categories  $y \in \mathcal{C}_{\text{large}}$ , because we believe the activations and the parameters have similar local and global structure in their respective space. Once this mapping has been learned on  $\mathcal{D}_{\text{large}}$ , because of this structure-preserving property, we expect it to generalize to categories in  $\mathcal{C}_{\text{few}}$ .

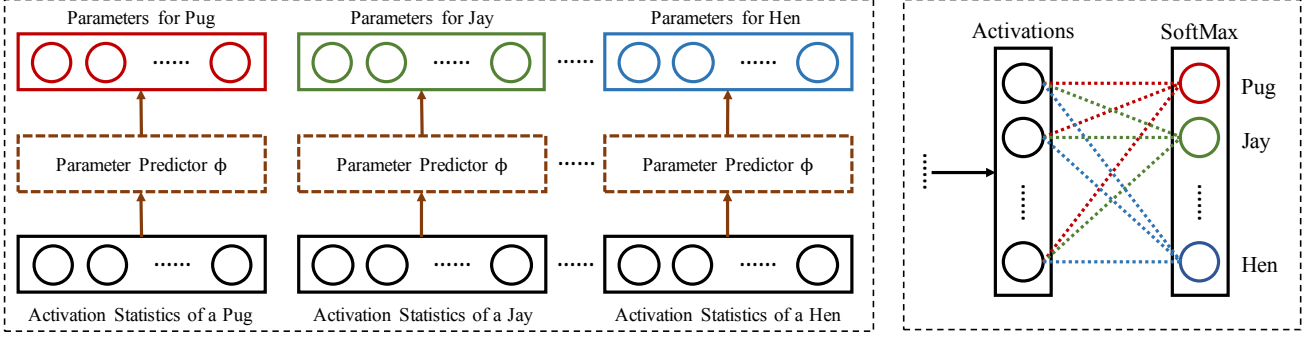


Figure 3: Building the fully connected layer by parameter prediction from activation statistics.

## 2.1. Learning Parameter Predictor

Since our final goal is to do classification, we learn  $\phi$  from the classification supervision. Specifically, we can learn  $\phi$  from  $\mathcal{D}_{\text{large}}$  by minimizing the classification loss (with a regularizer  $\|\phi\|$ ) defined by

$$\mathcal{L}(\phi) = \sum_{(y,x) \in \mathcal{D}_{\text{large}}} \left[ -\phi(\bar{\mathbf{a}}_y) \mathbf{a}(x) + \log \sum_{y' \in \mathcal{C}_{\text{large}}} e^{\phi(\bar{\mathbf{a}}_{y'}) \mathbf{a}(x)} \right] + \lambda \|\phi\| \quad (1)$$

Eq. 1 models the parameter prediction for categories  $y \in \mathcal{C}_{\text{large}}$ . However, for the few-shot set  $\mathcal{C}_{\text{few}}$ , each category only has a few activations, whose mean value is the activation itself when each category has only one sample. To model this few-shot setting in the large-scale training on  $\mathcal{D}_{\text{large}}$ , we allow both the individual activations and the mean activation to represent a category. Concretely, let  $\mathbf{s}_y \in \mathcal{A}_y \cup \bar{\mathbf{a}}_y$  be a statistic for category  $y$ . Let  $S_{\text{large}}$  denote a statistic set  $\{\mathbf{s}_1, \dots, \mathbf{s}_{|\mathcal{C}_{\text{large}}|}\}$  with one for each category in  $\mathcal{C}_{\text{large}}$ . We sample activations  $\mathbf{s}_y$  for each category  $y$  from  $\mathcal{A}_y \cup \bar{\mathbf{a}}_y$  with a probability  $p_{\text{mean}}$  to use  $\bar{\mathbf{a}}_y$  and  $1 - p_{\text{mean}}$  to sample uniformly from  $\mathcal{A}_y$ . Now, we learn  $\phi$  to minimize the loss defined by

$$\mathcal{L}(\phi) = \sum_{(y,x) \in \mathcal{D}_{\text{large}}} \mathbb{E}_{S_{\text{large}}} \left[ -\phi(\mathbf{s}_y) \mathbf{a}(x) + \log \sum_{y' \in \mathcal{C}_{\text{large}}} e^{\phi(\mathbf{s}_{y'}) \mathbf{a}(x)} \right] + \lambda \|\phi\| \quad (2)$$

## 2.2. Inference

During inference we include  $\mathcal{C}_{\text{few}}$ , which calls for a statistic set for all categories  $S = \{\mathbf{s}_1, \dots, \mathbf{s}_{|\mathcal{C}|}\}$ , where  $\mathcal{C} = \mathcal{C}_{\text{large}} \cup \mathcal{C}_{\text{few}}$ . Each statistic set  $S$  can generate a set of parameters  $\{\phi(\mathbf{s}_1), \dots, \phi(\mathbf{s}_{|\mathcal{C}|})\}$  that can be used for building a classifier on  $\mathcal{C}$ . Since we have more than one possible set  $S$  from the dataset  $\mathcal{D} = \mathcal{D}_{\text{large}} \cup \mathcal{D}_{\text{few}}$ , we can do classification based on all the possible  $S$ . Formally, we compute the probability of  $x$  being in category  $y$  by

$$P(y|x) = e^{\mathbb{E}_S[\phi(\mathbf{s}_y) \mathbf{a}(x)]} / \left( \sum_{y' \in \mathcal{C}} e^{\mathbb{E}_S[\phi(\mathbf{s}_{y'}) \mathbf{a}(x)]} \right) \quad (3)$$

However, classifying images with the above equation is time-consuming since it computes the expectations over the entire space of  $S$  which is exponentially large. We show in the following that if we assume  $\phi$  to be a linear mapping, then this expectation can be computed efficiently.

In the linear case  $\phi$  is a matrix  $\Phi$ . The predicted parameter for category  $y$  is

$$\hat{\mathbf{w}}_y = \Phi \cdot \mathbf{s}_y \quad (4)$$

The inner product of  $x$  before the softmax function for category  $y$  is

$$h(\mathbf{s}_y, \mathbf{a}(x)) = \hat{\mathbf{w}}_y \cdot \mathbf{a}(x) = \Phi \cdot \mathbf{s}_y \cdot \mathbf{a}(x) \quad (5)$$

If  $\mathbf{a}(x)$  and  $\mathbf{s}_y$  are normalized, then by setting  $\Phi$  as the identity matrix,  $h(\mathbf{s}_y, \mathbf{a}(x))$  is equivalent to the cosine similarity between  $\mathbf{s}_y$  and  $\mathbf{a}(x)$ . Essentially, by learning  $\Phi$ , we are learning a more general similarity metric on the activations  $\mathbf{a}(x)$  by capturing correlations between different dimensions of the activations. We will show more comparisons between the learned  $\Phi$  and identity matrix in §4.1.

Because of the linearity of  $\phi$ , the probability of  $x$  being in category  $y$  simplifies to

$$\begin{aligned} P(y|x) &= e^{\mathbf{a}(x) \cdot \phi(\mathbb{E}_S[\mathbf{s}_y])} / \left( \sum_{y' \in \mathcal{C}} e^{\mathbf{a}(x) \cdot \phi(\mathbb{E}_S[\mathbf{s}_{y'}])} \right) \\ &= e^{\mathbf{a}(x) \cdot \Phi \cdot \mathbb{E}_S[\mathbf{s}_y]} / \left( \sum_{y' \in \mathcal{C}} e^{\mathbf{a}(x) \cdot \Phi \cdot \mathbb{E}_S[\mathbf{s}_{y'}]} \right) \end{aligned} \quad (6)$$

Now  $\mathbb{E}_S[\mathbf{s}_y]$  can be pre-computed which is efficient. Adapting to novel categories only requires updating the corresponding  $\mathbb{E}_S[\mathbf{s}_y]$ . Although it is ideal to keep the linearity of  $\phi$  to reduce the amount of computation, introducing non-linearity could potentially improve the performance. To keep the efficiency, we still push in the expectation and approximate Eq. 3 as in Eq. 6.

When adding categories  $y \in \mathcal{C}_{\text{few}}$ , the estimate of  $\mathbb{E}_S[\mathbf{s}_y]$  may not be reliable since the number of samples is small. Besides, Eq. 2 models the sampling from one-shot and mean activations. Therefore, we take a mixed strategy for

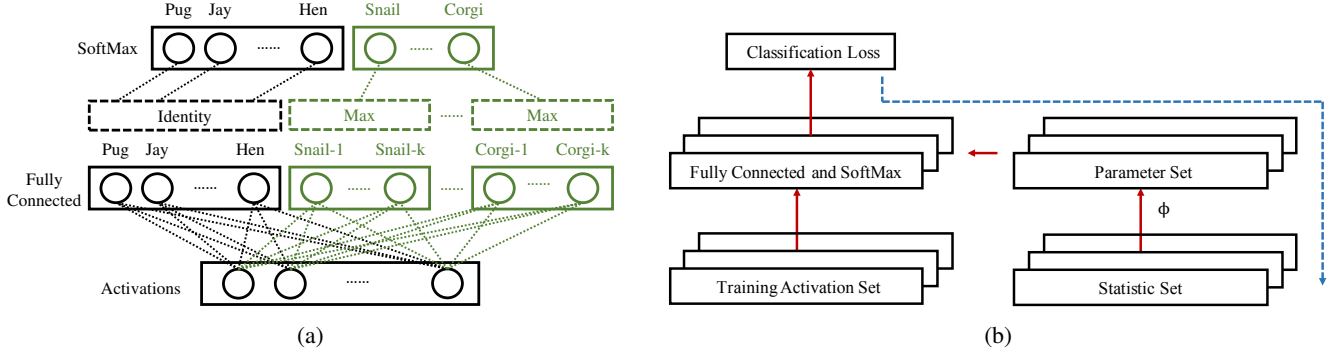


Figure 4: Illustration of the novel category adaption (a) and the training strategies for parameter predictor  $\phi$  (b). (b): red and solid arrows show the feedforward data flow, while blue and dashed arrow shows the backward gradient flow.

parameter prediction, *i.e.*, we use  $\mathbb{E}_S[s_y]$  to predict parameters for category  $y \in \mathcal{C}_{\text{large}}$ , but for  $\mathcal{C}_{\text{few}}$  we treat each sample as a newly added category, as shown in Figure 4a. For each novel category in  $\mathcal{C}_{\text{few}}$ , we compute the maximal response of the activation of the test image to the parameter set predicted from each activation in the statistic set of the corresponding novel category in  $\mathcal{C}_{\text{few}}$ . We use them as the inputs to the SoftMax layer to compute the probabilities.

### 2.3. Training Strategy

The objective of training is to find  $\phi$  that minimizes Eq. 2. There are many methods to do this. We approach this by using stochastic gradient descent with weight decay and momentum. Figure 4b demonstrates the training strategy of the parameter predictor  $\phi$ . We train  $\phi$  on  $\mathcal{D}_{\text{large}}$  with categories  $\mathcal{C}_{\text{large}}$ . For each batch of the training data, we sample  $|\mathcal{C}_{\text{large}}|$  statistics  $s_y$  from  $\mathcal{A}_y \cup \bar{\mathcal{a}}_y$  to build a statistic set  $S$  with one for each category  $y$  in  $\mathcal{C}_{\text{large}}$ . Next, we sample a training activation set  $T$  from  $\mathcal{D}_{\text{large}}$  with one for each category in  $\mathcal{C}_{\text{large}}$ . In total, we sample  $2|\mathcal{C}_{\text{large}}|$  activations. The activations in the statistic sets are fed to  $\phi$  to generate parameters for the fully connected layer. With the predicted parameters for each category in  $\mathcal{C}_{\text{large}}$ , the training activation set then is used to evaluate their effectiveness by classifying the training activations. At last, we compute the classification loss with respect to the ground truth, based on which we calculate the gradients and back-propagate them in the path shown in Figure 4b. After the gradient flow passes through  $\phi$ , we update  $\phi$  according to the gradients.

### 2.4. Implementation Details

**Full ImageNet Dataset** Our major experiments are conducted on ILSVRC 2015 [27]. ILSVRC 2015 is a large-scale image dataset with 1000 categories, each of which has about 1300 images for training, and 50 images for validation. For the purpose of studying both the large-scale and the few-shot settings at the same time, ILSVRC 2015 is split to two sets by the categories. The training data from

900 categories are collected into  $\mathcal{D}_{\text{large}}$ , while the rest 100 categories are gathered as set  $\mathcal{D}_{\text{few}}$ .

We first train a 50-layer ResNet [11] on  $\mathcal{D}_{\text{large}}$ . We use the outputs of the global average pooling layer as the activation  $\mathbf{a}(x)$  of an image  $x$ . For efficiency, we compute the activation  $\mathbf{a}(x)$  for each image  $x$  before the experiments as well as the mean activations  $\bar{\mathbf{a}}_y$ . Following the training strategy shown in §2.3, for each batch, we sample 900 activations as the statistic set and 900 activations as the training activation set. We compute the parameters using the statistic set, and copy the parameters into the fully connected layer. Then, we feed the training activations into the fully connected layer, calculate the loss and back-propagate the gradients. Next, we redirect the gradient flow into  $\phi$ . Finally, we update  $\phi$  using stochastic gradient descent. The learning rate is set to 0.001. The weight decay is set to 0.0005 and the momentum is set to 0.9. We train  $\phi$  on  $\mathcal{D}_{\text{large}}$  for 300 epochs, each of which has 250 batches.  $p_{\text{mean}}$  is set to 0.9.

For the parameter predictor, we implement three different  $\phi$ :  $\phi^1$ ,  $\phi^2$  and  $\phi^{2*}$ .  $\phi^1$  is a one-layer fully connected model.  $\phi^2$  is defined as a sequential network with two fully connected layers in which each maps from 2048 dimensional features to 2048 dimensional features and the first one is followed by a ReLU non-linearity layer [25]. The final outputs are normalized to unity in order to speed up training and ensure generalizability. By introducing non-linearity, we observe slight improvements on the accuracies for both  $\mathcal{C}_{\text{large}}$  and  $\mathcal{C}_{\text{few}}$ . To demonstrate the effect of minimizing Eq. 2 instead of Eq. 1, we train another  $\phi^{2*}$  which has the same architecture with  $\phi^2$  but minimizes Eq. 1. As we will show later through experiments,  $\phi^{2*}$  has strong bias towards  $\mathcal{C}_{\text{large}}$ .

**MiniImageNet Dataset** For comparison purposes, we also test our method on MiniImageNet dataset [29], a simplified subset of ILSVRC 2015. This dataset has 80 categories for  $\mathcal{D}_{\text{large}}$  and 20 categories for  $\mathcal{D}_{\text{few}}$ . Each category has 600 images. Each image is of size  $84 \times 84$ . For the



fairness of comparisons, we train two convolutional neural networks to get the activation functions  $\mathbf{a}(\cdot)$ . The first one is the same as that of Matching Network [29], and the second one is a wide residual network [33]. We train the wide residual network WRN-28-10 [33] on  $\mathcal{D}_{\text{large}}$ , following its configuration for CIFAR-100 dataset [15]. There are some minor modifications to the network architecture as the input size is different. To follow the architecture, the input size is set to  $80 \times 80$ . The images will be rescaled to this size before training and evaluation. There will be 3 times of downsampling rather than 2 times as for CIFAR dataset. The training process follows WRN-28-10 [33]. We also use the output of the global average pooling layer as the activation  $\mathbf{a}(x)$  of an image  $x$ . For the parameter predictor  $\phi$ , we train it by following the settings of  $\phi^2$  for the full ImageNet dataset except that now the dimension corresponds to the output of the activations of the convolutional neural networks. The two architectures will be detailed in the appendix.

### 3. Related Work

#### 3.1. Large-Scale Image Recognition

We have witnessed an evolution of image datasets over the last few decades. The sizes of the early datasets are relatively small. Each dataset usually collects images on the order of tens of thousands. Representative datasets include Caltech-101 [7], Caltech-256 [9], Pascal VOC [6], and CIFAR-10/100 [15]. Nowadays, large-scale datasets are available with millions of detailed image annotations, *e.g.* ImageNet [27] and MS COCO [21]. With datasets of this scale, machine learning methods that have large capacity start to prosper, and the most successful ones are convolutional neural network based [11, 12, 16, 28, 32].

#### 3.2. Few-Shot Image Recognition

Unlike large-scale image recognition, the research on few-shot learning has received limited attention from the community due to its inherent difficulty, thus is still at an early stage of development. As an early attempt, Fei-Fei *et al.* proposed a variational Bayesian framework for one-shot image classification [7]. A method called Hierarchical Bayesian Program Learning [19] was later proposed to specifically approach the one-shot problem on character recognition by a generative model. On the same character recognition task, Koch *et al.* developed a siamese convolutional network [14] to learn the representation from the dataset and modeled the few-shot learning as a verification task. Later, Matching Network [29] was proposed to approach the few-shot learning task by modeling the problem as a  $k$ -way  $m$ -shot image retrieval problem using attention and memory models. Following this work, Ravi and Larochelle proposed a LSTM-based meta-learner optimizer [26], and Chelsea *et al.* proposed a model-agnostic

meta learning method [8]. Although they show state-of-the-art performances on their few-shot learning tasks, they are not flexible for both large-scale and few-shot learning since  $k$  and  $m$  are fixed in their architectures. We will compare ours with these methods on their tasks for fair comparisons.

### 3.3. Unified Approach

Learning a metric then using nearest neighbor [14, 22, 30] is applicable but not necessarily optimal to the unified problem of large-scale and few-shot learning since it is possible to train a better model on the large-scale part of the dataset using the methods in §3.1. Mao *et al.* proposed a method called Learning like a Child [24] specifically for fast novel visual concept learning using hundreds of examples per category while keeping the original performance. However, this method is less effective when the training examples are extremely insufficient, *e.g.*  $< 6$  in this paper. Model Regression [31] and SGM [10] are suitable for both large-scale and few-shot image recognition. The comparisons with them are presented in the appendix.

## 4. Results

### 4.1. Full ImageNet Classification

In this section we describe our experiments and compare our approach with other strong baseline methods. As stated in §1, there are three aspects to consider in evaluating a method: (1) its performance on the few-shot set  $\mathcal{D}_{\text{few}}$ , (2) its performance on the large-scale set  $\mathcal{D}_{\text{large}}$ , and (3) its computation overhead of adding novel categories and the complexity of image inference. In the following paragraphs, we will cover the settings of the baseline methods, compare the performances on the large-scale and the few-shot sets, and discuss their efficiencies.

**Baseline Methods** The baseline methods must be applicable to both large-scale and few-shot learning settings. We compare our method with a fine-tuned 50-layer ResNet [11], Learning like a Child [24] with a pre-trained 50-layer ResNet as the starting network, Siamese-Triplet Network [14, 22] using three 50-layer ResNets with shared parameters, and the nearest neighbor using the pre-trained 50-layer ResNet convolutional features. We will elaborate individually on how to train and use them.

As mentioned in §2.4, we first train a 900-category classifier on  $\mathcal{D}_{\text{large}}$ . We will build other baseline methods using this classifier as the starting point. For convenience, we denote this classifier as  $\mathcal{R}_{\text{large}}^{\text{pt}}$ , where pt stands for “pre-trained”. Next, we add the novel categories  $\mathcal{C}_{\text{few}}$  to each method. For the 50-layer ResNet, we fine tune  $\mathcal{R}_{\text{large}}^{\text{pt}}$  with the newly added images by extending the fully connected layer to generate 1000 classification outputs. Note that we will limit the number of training samples of  $\mathcal{C}_{\text{few}}^{\text{pt}}$  for the few-shot setting. For Learning like a Child, however, we

Method	$\mathcal{D}_{\text{large}}$	$\mathcal{D}_{\text{few}}$	FT	Top-1 $\mathcal{C}_{\text{large}}$	Top-5 $\mathcal{C}_{\text{large}}$	Top-1 $\mathcal{C}_{\text{few}}$	Top-5 $\mathcal{C}_{\text{few}}$
NN + Cosine	100%	1	N	71.54%	91.20%	1.72%	5.86%
NN + Cosine	10%	1	N	67.68%	88.90%	4.42%	13.36%
NN + Cosine	1%	1	N	61.11%	85.11%	10.42%	25.88%
Triplet Network [14, 22]	100%	1	N	70.47%	90.61%	1.26%	4.94%
Triplet Network [14, 22]	10%	1	N	66.64%	88.42%	3.48%	11.40%
Triplet Network [14, 22]	1%	1	N	60.09%	84.83%	8.84%	22.24%
Fine-Tuned ResNet [11]	100%	1	Y	76.28%	93.17%	2.82%	13.30%
Learning like a Child [24]	100%	1	Y	76.71%	93.24%	2.90%	17.14%
Ours- $\phi^1$	100%	1	N	72.56%	91.12%	<b>19.88%</b>	<b>43.20%</b>
Ours- $\phi^2$	100%	1	N	74.17%	91.79%	<b>21.58%</b>	<b>45.82%</b>
Ours- $\phi^{2*}$	100%	1	N	75.63%	92.92%	14.32%	33.84%
NN + Cosine	100%	2	N	71.54%	91.20%	3.34%	9.88%
NN + Cosine	10%	2	N	67.66%	88.89%	7.60%	19.94%
NN + Cosine	1%	2	N	61.04%	85.04%	15.14%	35.70%
Triplet Network [14, 22]	100%	2	N	70.47%	90.61%	2.34%	8.30%
Triplet Network [14, 22]	10%	2	N	66.63%	88.41%	6.10%	17.46%
Triplet Network [14, 22]	1%	2	N	60.02%	84.74%	13.42%	32.38%
Fine-Tuned ResNet [11]	100%	2	Y	76.27%	93.13%	10.32%	30.34%
Learning like a Child [24]	100%	2	Y	76.68%	93.17%	11.54%	37.68%
Ours- $\phi^1$	100%	2	N	71.94%	90.62%	<b>25.54%</b>	<b>52.98%</b>
Ours- $\phi^2$	100%	2	N	73.43%	91.13%	<b>27.44%</b>	<b>55.86%</b>
Ours- $\phi^{2*}$	100%	2	N	75.44%	92.74%	18.70%	43.92%
NN + Cosine	100%	3	N	71.54%	91.20%	4.58%	12.72%
NN + Cosine	10%	3	N	67.65%	88.88%	9.86%	24.96%
NN + Cosine	1%	3	N	60.97%	84.95%	18.68%	42.16%
Triplet Network [14, 22]	100%	3	N	70.47%	90.61%	3.22%	11.48%
Triplet Network [14, 22]	10%	3	N	66.62%	88.40%	8.52%	22.52%
Triplet Network [14, 22]	1%	3	N	59.97%	84.66%	17.08%	38.06%
Fine-Tuned ResNet [11]	100%	3	Y	76.25%	93.07%	16.76%	39.92%
Learning like a Child [24]	100%	3	Y	76.55%	93.00%	18.56%	50.70%
Ours- $\phi^1$	100%	3	N	71.56%	90.21%	<b>28.72%</b>	<b>58.50%</b>
Ours- $\phi^2$	100%	3	N	72.98%	90.59%	<b>31.20%</b>	<b>61.44%</b>
Ours- $\phi^{2*}$	100%	3	N	75.34%	92.60%	22.32%	49.76%

Table 1: Comparing 1000-way accuracies with feature extractor  $a(\cdot)$  pre-trained on  $\mathcal{D}_{\text{large}}$ . For different  $\mathcal{D}_{\text{few}}$  settings, red: the best few-shot accuracy, and blue: the second best.

fix the layers before the global average pooling layer, extend the fully connected layer to include 1000 classes, and only update the parameters for  $\mathcal{C}_{\text{few}}$  in the last classification layer. Since we have the full access to  $\mathcal{D}_{\text{large}}$ , we do not need Baseline Probability Fixation [24]. The nearest neighbor with cosine distance can be directly used for both tasks given the pre-trained deep features.

The other method we compare is Siamese-Triplet Network [14, 22]. Siamese network is proposed to approach the few-shot learning problem on Omniglot dataset [17]. In our experiments, we find that its variant Triplet Network [22, 30] is more effective since it learns feature representation from relative distances between positive and negative pairs instead of directly doing binary classification from the feature distance. Therefore, we use the Triplet Network from [22] on the few-shot learning problem, and upgrade its body net to the pre-trained  $\mathcal{R}_{\text{large}}^{\text{pt}}$ . We use cosine distance as the distance metric and fine-tune the Triplet Network. For inference, we use nearest neighbor with cosine distance. We use some techniques to improve the speed, which will be discussed later in the efficiency analysis.

**Few-Shot Accuracies** We first investigate the few-shot learning setting where we only have several training examples for  $\mathcal{C}_{\text{few}}$ . Specifically, we study the performances of different methods when  $\mathcal{D}_{\text{few}}$  has for each category 1, 2, and 3 samples. It is worth noting that our task is much harder than the previously studied few-shot learning: we are evaluating the top predictions out of 1000 candidate categories, *i.e.*, 1000-way accuracies while previous work is mostly interested in 5-way or 20-way accuracies [8, 14, 22, 26, 29].

With the pre-trained  $\mathcal{R}_{\text{large}}^{\text{pt}}$ , the training samples in  $\mathcal{D}_{\text{few}}$  are like invaders to the activation space for  $\mathcal{C}_{\text{large}}$ . Intuitively, there will be a trade-off between the performances on  $\mathcal{C}_{\text{large}}$  and  $\mathcal{C}_{\text{few}}$ . This is true especially for non-parametric methods. Table 1 shows the performances on the validation set of ILSVRC 2015 [27]. The second column is the percentage of data of  $\mathcal{D}_{\text{large}}$  in use, and the third column is the number of samples used for each category in  $\mathcal{D}_{\text{few}}$ . Note that fine-tuned ResNet [11] and Learning like a Child [24] require fine-tuning while others do not.

Triplet Network is designed to do few-shot image inference by learning feature representations that adapt to the



Classifier	Top-1 $C_{\text{large}}$	Top-5 $C_{\text{large}}$	Top-1 $C_{\text{few}}$	Top-5 $C_{\text{few}}$
NN	70.25%	89.98%	52.46%	80.94
Linear	75.20%	92.38%	60.50%	87.58

Table 2: Oracle 1000-way accuracies of the feature extractor  $\mathbf{a}(\cdot)$  pre-trained on  $\mathcal{D}_{\text{large}}$ .

chosen distance metric. It has better performance on  $C_{\text{few}}$  compared with the fine-tuned ResNet and Learning like a Child when the percentage of  $\mathcal{D}_{\text{large}}$  in use is low. However, its accuracies on  $C_{\text{large}}$  are sacrificed a lot in order to favor few-shot accuracies. We also note that if full category supervision is provided, the activations of training a classifier do better than that of training a Triplet Network. We speculate that this is due to the less supervision of training a Triplet Network which uses losses based on fixed distance preferences. Fine-tuning and Learning like a Child are training based, thus are able to keep the high accuracies on  $\mathcal{D}_{\text{large}}$ , but perform badly on  $\mathcal{D}_{\text{few}}$  which does not have sufficient data for training. Compared with them, our method shows state-of-the-art accuracies on  $C_{\text{few}}$  without compromising too much the performances on  $C_{\text{large}}$ .

Table 1 also compares  $\phi^2$  and  $\phi^{2*}$ , which are trained to minimize Eq. 2 and Eq. 1, respectively. Since during training  $\phi^{2*}$  only mean activations are sampled, it shows a bias towards  $C_{\text{large}}$ . However, it still outperforms other baseline methods on  $C_{\text{few}}$ . In short, modeling using Eq. 2 and Eq. 1 shows a tradeoff between  $C_{\text{large}}$  and  $C_{\text{few}}$ .

**Oracles** Here we explore the upper bound performance on  $C_{\text{few}}$ . In this setting we have all the training data for  $C_{\text{large}}$  and  $C_{\text{few}}$  in ImageNet. For the fixed feature extractor  $\mathbf{a}(\cdot)$  pre-trained on  $\mathcal{D}_{\text{large}}$ , we can train a linear classifier on  $C_{\text{large}}$  and  $C_{\text{few}}$ , or use nearest neighbor, to see what are the upper bounds of the pre-trained  $\mathbf{a}(\cdot)$ . Table 2 shows the results. The performances are evaluated on the validation set of ILSVRC 2015 [27] which has 50 images for each category. The feature extractor pre-trained on  $\mathcal{D}_{\text{large}}$  demonstrates reasonable accuracies on  $C_{\text{few}}$  which it has never seen during training for both parametric and non-parametric methods.

**Efficiency Analysis** We briefly discuss the efficiencies of each method including ours on the adaptation to novel categories and the image inference. The methods are tested on NVIDIA Tesla K40M GPUs. For adapting to novel categories, fine-tuned ResNet and Learning like a Child require re-training the neural networks. For re-training one epoch of the data, fine-tuned ResNet and Learning like a Child both take about 1.8 hours on 4 GPUs. Our method only needs to predict the parameters for the novel categories using  $\phi$  and add them to the original neural network. This process takes 0.683s using one GPU for adapting the network

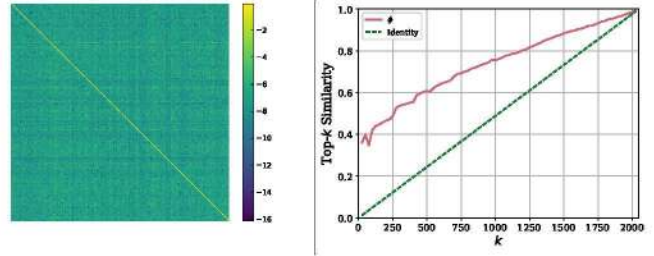


Figure 5: Visualization of the upper-left  $256 \times 256$  submatrix of  $\phi^1$  in log scale (left) and top- $k$  similarity between  $\phi^1$ ,  $\mathbf{1}$  and  $\mathbf{w}_{\text{large}}^{\text{pt}}$  (right). In the right plotting, red and solid lines are similarities between  $\phi^1$  and  $\mathbf{w}_{\text{large}}^{\text{pt}}$ , and green and dashed lines are between  $\mathbf{1}$  and  $\mathbf{w}_{\text{large}}^{\text{pt}}$ .

to 100 novel categories with one example each. Siamese-Triplet Network and nearest neighbor with cosine distance require no operations for adapting to novel categories as they are ready for feature extraction.

For image inference, Siamese-Triplet Network and nearest neighbor are very slow since they will look over the entire dataset. Without any optimization, this can take 2.3 hours per image when we use the entire  $\mathcal{D}_{\text{large}}$ . To speed up this process in order to do comparison with ours, we first pre-compute all the features. Then, we use a deep learning framework to accelerate the cosine similarity computation. At the cost of 45GB memory usage and the time for feature pre-computation, we manage to lower the inference time of them to 37.867ms per image. Fine-tuned ResNet, Learning like a Child and our method are very fast since at the inference stage, these three methods are just normal deep neural networks. The inference speed of these methods is about 6.83ms per image on one GPU when the batch size is set to 32. In a word, compared with other methods, our method is fast and efficient in both the novel category adaptation and the image inference.

**Comparing Activation Impacts** In this subsection we investigate what  $\phi^1$  has learned that helps it perform better than the cosine distance, which is a special solution for one-layer  $\phi$  by setting  $\phi$  to the identity matrix  $\mathbf{1}$ . We first visualize the matrix  $\phi_{i,j}^1$  in log scale as shown in the left image of Figure 5. Due to the space limit, we only show the upper-left  $256 \times 256$  submatrix. Not surprisingly, the values on the diagonal dominates the matrix. We observe that along the diagonal, the maximum is 0.976 and the minimum is 0.744, suggesting that different from  $\mathbf{1}$ ,  $\phi^1$  does not use each activation channel equally. We speculate that this is because the pre-trained activation channels have different distributions of magnitudes and different correlations with the classification task. These factors can be learned by the last fully connected layer of  $\mathcal{R}_{\text{large}}^{\text{pt}}$  with large amounts of data but are

Method	1-Shot	5-Shot
Fine-Tuned Baseline	28.86 ± 0.54%	49.79 ± 0.79%
Nearest Neighbor	41.08 ± 0.70%	51.04 ± 0.65%
Matching Network [29]	43.56 ± 0.84%	55.31 ± 0.73%
Meta-Learner LSTM [26]	43.44 ± 0.77%	60.60 ± 0.71%
MAML [8]	48.70 ± 1.84%	63.11 ± 0.92%
Ours-Simple	<b>54.53</b> ± 0.40%	<b>67.87</b> ± 0.20%
Ours-WRN	<b>59.60</b> ± 0.41%	<b>73.74</b> ± 0.19%

Table 3: 5-way accuracies on MiniImageNet with 95% confidence interval. Red: the best, and blue: the second best.

assumed equal for every channel in cosine distance. This motivates us to investigate the impact of each channel of the activation space.

For a fixed activation space, we define the *impact* of its  $j$ -th channel on mapping  $\phi$  by  $I_j(\phi) = \sum_i |\phi_{ij}|$ . Similarly, we define the activation impact  $I_j(\cdot)$  on  $\mathbf{w}_{\text{large}}^{\text{pt}}$  which is the parameter matrix of the last fully connected layer of  $\mathcal{R}_{\text{large}}^{\text{pt}}$ . For cosine distance,  $I_j(\mathbf{1}) = 1, \forall j$ . Intuitively, we are evaluating the impact of each channel of  $\phi$  on the output by adding all the weights connected to it. For  $\mathbf{w}_{\text{large}}^{\text{pt}}$  which is trained for the classification task using large-amounts of data, if we normalize  $I(\mathbf{w}_{\text{large}}^{\text{pt}})$  to unity, the mean of  $I(\mathbf{w}_{\text{large}}^{\text{pt}})$  over all channel  $j$  is 2.13e-2 and the standard deviation is 5.83e-3.  $\mathbf{w}_{\text{large}}^{\text{pt}}$  does not use channels equally, either.

In fact,  $\phi^1$  has a high similarity with  $\mathbf{w}_{\text{large}}^{\text{pt}}$ . We show this by comparing the orders of the channels sorted by their impacts. Let  $\text{top-}k(S)$  find the indexes of the top- $k$  elements of  $S$ . We define the top- $k$  similarity of  $I(\phi)$  and  $I(\mathbf{w}_{\text{large}}^{\text{pt}})$  by

$$\text{OS}(\phi, \mathbf{w}_{\text{large}}^{\text{pt}}, k) = \text{card}(\text{top-}k(I(\phi)) \cap \text{top-}k(I(\mathbf{w}_{\text{large}}^{\text{pt}}))) / k \quad (7)$$

where  $\text{card}$  is the cardinality of the set. The right image of Figure 5 plots the two similarities, from which we observe high similarity between  $\phi$  and  $\mathbf{w}_{\text{large}}^{\text{pt}}$  compared to the random order of  $\mathbf{1}$ . From this point of view,  $\phi^1$  outperforms the cosine distance due to its better usage of the activations.

## 4.2. MiniImageNet Classification

In this subsection we compare our method with the previous state-of-the-arts on the MiniImageNet dataset. Unlike ImageNet classification, the task of MiniImageNet is to find the correct category from 5 candidates, each of which has 1 example or 5 examples for reference. The methods are only evaluated on  $D_{\text{few}}$ , which has 20 categories. For each task, we uniformly sample 5 categories from  $D_{\text{few}}$ . For each of the category, we randomly select one or five images as the references, depending on the settings, then regard the rest images of the 5 categories as the test images. For each task, we will have an average accuracy over this 5 categories. We repeat the task with different categories and report the mean of the accuracies with the 95% confidence interval.

Table 3 summarizes the few-shot accuracies of our method and the previous state-of-the-arts. For fair comparisons, we implement two convolutional neural networks. The convolutional network of Ours-Simple is the same as that of Matching Network [29] while Ours-WRN uses WRN-28-10 [33] as stated in §2.4. The experimental results demonstrate that our average accuracies are better than the previous state-of-the-arts by a large margin for both the Simple and WRN implementations.

It is worth noting that the methods [8, 26, 29] are not evaluated in the full ImageNet classification task. This is because the architectures of these methods, following the problem formulation of Matching Network [29], can only deal with the test tasks that are of the same number of reference categories and images as that of the training tasks, limiting their flexibilities for classification tasks of arbitrary number of categories and reference images. In contrast, our proposed method has no assumptions regarding the number of the reference categories and the images, while achieving good results on both tasks. From this perspective, our methods are better than the previous state-of-the-arts in terms of both the performance and the flexibility.

## 5. Conclusion

In this paper, we study a novel problem: can we develop a unified approach that works for both large-scale and few-shot learning. Our motivation is based on the observation that in the final classification layer of a pre-trained neural network, the parameter vector and the activation vector have highly similar structures in space. This motivates us to learn a category-agnostic mapping from activations to parameters. Once this mapping is learned, the parameters for any novel category can be predicted by a simple forward pass, which is significantly more convenient than re-training used in parametric methods or enumeration of training set used in non-parametric approaches.

We experiment our novel approach on the MiniImageNet dataset and the challenging full ImageNet dataset. The challenges of the few-shot learning on the full ImageNet dataset are from the large number of categories (1000) and the very limited number ( $< 4$ ) of training samples for  $\mathcal{C}_{\text{few}}$ . On the full ImageNet dataset, we show promising results, achieving state-of-the-art classification accuracy on novel categories by a significant margin while maintaining comparable performance on the large-scale classes. On the small MiniImageNet dataset, we also outperform the previous state-of-the-art methods by a large margin. The experimental results demonstrate the effectiveness of the proposed method for learning a category-agnostic mapping.

**Acknowledgments** We gratefully acknowledge funding supports from NSF CCF-1231216 and ONR N00014-15-1-2356. This work was also supported by the National Natural Science Foundation of China under Grant 61672336.



## References

- [1] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning for control. In *Lazy learning*, pages 75–113. Springer, 1997. **1**
- [2] P. Bloom. *How children learn the meanings of words*, volume 377. Citeseer, 2000. **1**
- [3] Z. Chen and B. Liu. Mining topics in documents: standing on the shoulders of big data. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 1116–1125, 2014. **1**
- [4] Z. Chen and B. Liu. Topic modeling using topics from many domains, lifelong learning and big data. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 703–711, 2014. **1**
- [5] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 647–655, 2014. **2**
- [6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. <http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html>. **5**
- [7] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006. **1, 5**
- [8] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning Research*, pages 1126–1135, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. **5, 6, 8**
- [9] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. In *Technical Report 7694, California Institute of Technology*, 2007. **5**
- [10] B. Hariharan and R. B. Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 3037–3046, 2017. **5**
- [11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. **1, 2, 4, 5, 6**
- [12] G. Huang, Z. Liu, and K. Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. **5**
- [13] M. Huh, P. Agrawal, and A. A. Efros. What makes imagenet good for transfer learning? *CoRR*, abs/1608.08614, 2016. **2**
- [14] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning workshop*, 2015. **1, 5, 6**
- [15] A. Krizhevsky. Learning multiple layers of features from tiny images. In *Masters thesis, Department of Computer Science, University of Toronto*, 2009. **5**
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. **1, 5**
- [17] B. M. Lake, R. Salakhutdinov, J. Gross, and J. B. Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the 33th Annual Meeting of the Cognitive Science Society, CogSci 2011, Boston, Massachusetts, USA, July 20-23, 2011*. **6**
- [18] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015. **1**
- [19] B. M. Lake, R. R. Salakhutdinov, and J. Tenenbaum. One-shot learning by inverting a compositional causal process. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2526–2534. Curran Associates, Inc., 2013. **5**
- [20] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. **1**
- [21] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. **5**
- [22] X. Lin, H. Wang, Z. Li, Y. Zhang, A. Yuille, and T. S. Lee. Transfer of view-manifold learning to similarity perception of novel objects. In *5th International Conference on Learning Representations (ICLR)*, 2017. **1, 5, 6**
- [23] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008. **2**
- [24] J. Mao, X. Wei, Y. Yang, J. Wang, Z. Huang, and A. L. Yuille. Learning like a child: Fast novel visual concept learning from sentence descriptions of images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2533–2541, 2015. **1, 5, 6**
- [25] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814, 2010. **4**
- [26] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *5th International Conference on Learning Representations (ICLR)*, 2017. **5, 6, 8**
- [27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. **1, 2, 4, 5, 6, 7**
- [28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. **1, 5**
- [29] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In

*Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3630–3638, 2016. [1](#), [2](#), [4](#), [5](#), [6](#), [8](#)

- [30] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2794–2802, 2015. [5](#), [6](#)
- [31] Y. Wang and M. Hebert. Learning to learn: Model regression networks for easy small sample learning. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI*, pages 616–634, 2016. [5](#)
- [32] Y. Wang, L. Xie, C. Liu, Y. Zhang, W. Zhang, and A. Yuille. Sort: Second-order response transform for visual recognition. *CoRR*, abs/1703.06993, 2017. [5](#)
- [33] S. Zagoruyko and N. Komodakis. Wide residual networks. In *BMVC*, 2016. [5](#), [8](#)