Coarsely-labeled Data for Better Few-shot Transfer

Cheng Perng Phoo, Bharath Hariharan
Cornell University
{cpphoo, bharathh}@cs.cornell.edu

Abstract

Few-shot learning is based on the premise that labels are expensive, especially when they are fine-grained and require expertise. But coarse labels might be easy to acquire and thus abundant. We present a representation learning approach - PAS that allows few-shot learners to leverage coarsely-labeled data available before evaluation. Inspired by self-training, we label the additional data using a teacher trained on the base dataset and filter the teacher’s prediction based on the coarse labels; a new student representation is then trained on the base dataset and the pseudo-labeled dataset. PAS is able to produce a representation that consistently and significantly outperforms the baselines in 3 different datasets. Code is available at https://github.com/cpphoo/PAS

1. Introduction

Large annotated datasets [4, 8, 19] have empowered the progress of visual recognition systems over the last decade. However, for many practically important recognition problems, annotations might require expertise and thus might be difficult to acquire. For example, to build a recognition system that identifies insect species, one would have to hire an entomologist to label hundreds of thousands of images from hundreds of species: an expensive, time-consuming affair.

This concern has sparked research in few-shot learning (FSL), which aims to train domain-specific learners that can learn new classes from very few examples. These learners are “meta-trained” on a large labeled dataset of “base” classes from the same domain. The hope is that this base dataset provides the learner with the right inductive biases for the domain of interest so that recognizing “novel” classes does not require quite as much labeled data. FSL is now an extremely active research area with a veritable array of recent results [33, 6, 12, 10, 16, 44, 38, 47, 30, 23]. Yet, existing FSL systems still lag far behind systems trained with large quantities of labeled training data. One might conjecture that the base dataset does not provide sufficient information about the novel classes.

One possible approach to address this issue is to leverage some auxiliary information about novel classes that might be more readily available. For example, recent work [26, 9, 35] uses unlabeled data from the novel classes: it is, after all, the labels that are expensive; data is often cheap. While such unlabeled data can inform the learner about the data distribution of the novel classes, they contain no information about the semantics of the class distinctions.

A potential source of auxiliary information about semantics is labeling at a coarser granularity, which might be easier to obtain than the actual labels of interest. Consider again the problem of insect classification. It is true that one would have to hire an entomologist or even a lepidopterist to help distinguish between the 3 species of butterflies in figure 1; these labels are therefore difficult to acquire. But a layperson would be able to distinguish between butterflies and bees. Labels at that coarse granularity can thus be crowdsourced quite easily. This leads us to the following question: what if we had access to data from the novel classes that were weakly labeled with easy-to-acquire coarse labels?

Although such coarsely-labeled data are both readily available and potentially informative, no current FSL tech-
nique is capable of using this extra information. Class taxonomies have been explored in traditional supervised learning through hierarchical inference strategies, but it is unclear if these address the few-shot generalization problem. One could use these additional labels as an auxiliary loss in a multitask-training framework. However, it has been shown that multitask-training is not guaranteed to help all tasks [34]. Besides, multitask-training ignores the strong constraints that tie the coarse and fine labels together, thus missing out on vital semantic knowledge.

We propose a new few-shot learning approach that effectively leverages coarsely-labeled data. Following recent results, we focus on improving the feature representation, since this turns out to be crucial for FSL [38, 2, 11]. Inspired from recent work based on pseudo-labeling and self-training [26], we develop a representation learning approach named Parent-Aware Self-training (PAS). Specifically, we use a classifier trained on the base dataset to provide fine-grained pseudo-labels to coarsely-labeled data. These pseudo-labels are filtered so that they are consistent with the coarse labels. These pseudo-labels will definitely be incorrect, because they will wrongly declare novel class examples to be from one of the base classes. However, they will induce a fine-grained grouping of the coarsely-labeled novel examples that is consistent with other fine-grained base classes with the same coarse labels. We then train with these pseudo-labels to produce a feature representation that hopefully captures the unknown novel class distinctions.

We experiment with three different datasets, and compare representations that use such coarsely-labeled data and those that do not. We find that using coarsely-labeled data improves five-shot accuracy between 5 to 15 points on the challenging all-way classification setup. Our particular approach is also the best way to use this additional data, providing up to an average (across datasets) of 2 points improvement in five-shot accuracy compared to multitask training. All these gains vindicate the power of this additional information and the ability of our approach to use it.

2. Related Work

Few-shot Learning (FSL). We tackle FSL in our work. There are three main categorizations of FSL techniques: initialization-based approaches [6, 7, 28, 24, 31, 36, 16] build model initializations that can lead to rapid convergence on the base classes, positing that such initializations can also be good model initializations for the novel classes; metric learning approaches [41, 33, 37, 14, 10, 27] build a metric on the base dataset, assuming that base and novel share similar discriminative features; augmentation-based approaches [12, 43, 3] aim to learn augmentation mechanisms on the base dataset, postulating that base and novel classes share some class agnostic, intra-class variations. Most FSL techniques assume no access to the data from the novel classes when training the learners and solely hinge on the similarity between base and novel datasets. This critical assumption has led to underperformance of FSL techniques when the gap between base and novel datasets are large [11, 2]. To remedy this, we propose to use easy-to-acquire coarsely-labeled data from the novel classes in FSL.

FSL with Additional Data. Additional data have proven to be useful in FSL. The two most common setup that utilizes additional data are: semi-supervised FSL [29, 18, 48, 30, 42] and transductive FSL [23, 5]. Different from our setup, the additional data in these setups are unlabeled and only available during evaluation. [20] operates in a setup similar to ours but they focus on developing specialized inference procedures using the class hierarchy. In particular, they build upon Prototypical Network [33] and seek to build learned prototypes for coarse classes during representation learning that can be used to refine the prototypes for fine-grained classes during evaluation. In contrast, our approach focuses on building representations that are agnostic to any inference methods. The two approaches focus on different aspects of the setup and can be combined.

FSL with Class Taxonomy. Leveraging class taxonomy or hierarchy is common in supervised machine learning [1]. In fact, a survey on hierarchical classification by Silla and Freitas [32] have shown that in a wide range of application domains, incorporating class hierarchy when building classifiers can yield performance gains. In FSL, class taxonomy has been used to build better techniques. Efforts include building specific ConvNet architectures using semantic relationships between base and novel classes [17, 25] or specialized inference procedures that leverage the class taxonomy [21, 22]. All these methods can be directly used in our setup but they do not consider the use of coarsely-labeled examples which could bring forth more improvements.

Self-training. Our approach for using coarsely-labeled data is closely related to self-training. Self-training is often used in semi-supervised learning. The idea is to use a teacher model trained on the labeled data to label the unlabeled data and train another student model on both the original labeled data and the pseudo-labeled data. This simple technique has been shown to improve ImageNet classification performance [45, 46]. Another venue where self-training is used is knowledge distillation where the goal is to compress a large teacher model by training a student to reproduce the teacher’s predictions. Self-training has also been used in semi-supervised FSL [18, 42]. However, most of these approaches deploy self-training in a “closed set” setup, i.e., the set of classes is fixed, and the unlabeled data comes from this same set of classes. Thus, there is a significant chance the pseudo-labels are actually correct (though [45] do note that noisy pseudo-labels help).

Our approach moves away from the closed set setup and labels the coarsely-labeled novel class data with fine-
grained base class pseudo-labels that are guaranteed to be incorrect: an uncommon scenario. The only related work here is [26] in which the authors adapt feature representations to far off domains by training a network to replicate pseudo-labels produced by an unrelated classifier from a distant source domain. The authors observe that this is effective if the groupings induced by the pseudo-labels match class distinctions in the novel domain. However, the authors only use unlabeled data during representation learning resulting in potentially poor pseudo-labels, and as such resort to additional tricks such as self-supervised learning techniques. In contrast, our coarsely-labeled data leads to much better pseudo-labels, removing the need for such tricks.

3. Problem Setup

Our setup is illustrated in figure 2. We assume that we have a taxonomy of classes with two levels, a set of fine-grained classes that are more challenging to annotate (C) and a set of coarse-grained classes that are easier to annotate (P). The classes we are interested in recognizing are the former. Every fine-grained class c is associated to a single coarse-grained class p(c), i.e., the taxonomy is a tree. The fine-grained classes are split into base classes C_base and novel classes C_novel. Similar to the traditional few-shot classification setup, the goal is to build learners that can quickly learn to recognize novel classes C_novel each of which has very few training images.

Before encountering the novel classes, the learner fits its parameters in a representation learning phase. In this phase, similar to FSL, we assume that the learner has access to a large annotated base dataset D_fine:

$$D_{\text{rep}}^{\text{fine}} = \{(x_i, y_i, p_i)\}_{i=1}^n$$  \hspace{1cm} (1)

where x_i is the image, y_i is the base class label and p_i is the coarse label associated to base class y_i. Different from conventional FSL, we assume that the learner has access to an additional set of coarsely-labeled examples D_coarse:

$$D_{\text{rep}}^{\text{coarse}} = \{(x_j, p_j)\}_{j=1}^{n'}$$  \hspace{1cm} (2)

that contains images x_j from some of the novel classes but weakly annotated with coarse label p_j.

We define the representation set as D_rep = D_fine \cup D_coarse. We assume that we know the parents for the base classes, so that D_fine can also be decorated with coarse labels. We also assume that only a subset of the novel classes, C_novel seen, are “seen” by the learner through D_coarse (with only coarse labels). The remainder of the novel classes are “unseen” C_novel unseen = C_novel - C_novel seen.

After the representation learning phase, the learner goes into the evaluation phase where it gets a small reference set D_ref = \{(x_j, y_j)\}_{j=1}^{n_{ref}} of novel class examples x_j and their corresponding label y_j. In our experiments, D_ref is disjoint from D_rep though this is not necessary. Using D_ref, the learner must train a classifier for the novel classes, which will be evaluated on a completely unseen, unlabeled query set of novel class examples D_query:

We stress that the coarse label of reference and query examples are not revealed during evaluation.

For most of our experiments, we assume that each novel class has a base class as its sibling in the taxonomy. We explore the scenario where some novel classes are not related to any base classes in section 6.2.2.

4. Methodology

The goal is to build learners that can output a classification model f_0 parametrized by \theta upon receiving a small \text{D}_{\text{ref}}. We assume f_0 consists of two components: a feature extractor f_{\phi_0} that maps an input image x into \mathbb{R}^d and a classification model h_0(\cdot) that maps \phi_0(x) to the predicted probabilities \mathbb{P}_\theta(y|x). In general, the feature extractor \phi_0 would be learned during representation learning and kept fixed during few-shot evaluation to avoid overfitting.

4.1. Parent-Aware Self-training, PAS

We learn our feature representation \phi_0, as follows:

1. Learn a teacher model f_{\theta_0} on D_fine via minimizing cross entropy loss (with respect to the base classes).
2. Use the teacher model f_{\theta_0} to “pseudo-label” the coarsely-labeled dataset D_coarse. Crucially, we use the
coarse labels to filter the pseudo-labels:

\[ D_{\text{rep}}^{\text{pseudo}} = \{ (x_j, p_j, \bar{y}_j) \}_{j=1}^{n'} \] (3)

\[ \bar{y}_j = g(f_{\theta_0}(x_j), p_j) \quad \forall (x_j, p_j) \in D_{\text{rep}}^{\text{coarse}} \] (4)

where \( g \) filters the pseudo-labels \( f_{\theta_0}(x_j) \) based on the coarse-label \( p_j \) (section 4.1.1).

3. Learn a new student model \( f_{\theta} \) on \( D_{\text{rep}}^{\text{base}} \) and \( D_{\text{rep}}^{\text{pseudo}} \):

\[
\min_{\theta} \frac{1}{n} \sum_{(x_i, y_i) \in D_{\text{rep}}^{\text{fine}}} l_{CE}(f_\theta(x_i), y_i) + \frac{1}{n'} \sum_{(x_j, \bar{y}_j) \in D_{\text{rep}}^{\text{pseudo}}} l_{KL}(f_\theta(x_j), \bar{y}_j) \tag{5}
\]

where \( l_{CE} \) is the cross entropy loss, \( l_{KL} \) is the Kullback-Leibler divergence.

After representation learning, the student’s feature extractor \( \phi_{\theta^*} \) can be used to extract features for training the downstream classifier on the reference set.

### 4.1.1 Filtering Function \( g \)

Consider a data point \( x \) with coarse label \( p \). The pseudo-labels produced by the teacher \( \mathbb{P}_{\theta_0}(y=k|x) = f_{\theta_0}(x) \) need not be consistent with the coarse labels. This is especially true for the coarsely-labeled novel class examples, since these are sampled from a different data distribution as compared to the base classes. We therefore filter the pseudo-labels to encourage consistency between the pseudo-labels \( f_{\theta_0}(x) \) and the coarse labels \( p \). To do so, we first zero out the predicted probabilities for fine-grained labels that are inconsistent with \( p \) to produce an unnormalized probability vector \( \bar{s} \):

\[
\bar{s}[k] = \begin{cases} 0 & \text{if } p(k) \neq p \\ \mathbb{P}_{\theta_0}(y=k|x) & \text{otherwise} \end{cases} \tag{6}
\]

We then renormalize \( \bar{s}_j \) to construct the filtered soft pseudo-label:

\[ g(f_{\theta_0}(x), p) = \frac{\bar{s}}{\sum_k \bar{s}[k]} \tag{7} \]

Intuitively, the filtering function ensures that an example with coarse label \( p \) would only have non-zero probability mass for base classes associated to coarse label \( p \).

### 4.2. Inference Strategy

During evaluation, a variety of inference methods [33, 10] can be used along with the student’s representation during inference. For simplicity, we decided to use classifiers based on the nearest class prototype [33]. For each class \( k \) we compute the class prototype:

\[ \bar{c}_k = \frac{1}{\sum_j \mathbb{1}[y_j = k]} \sum_{x_j \in D_{\text{rep}}; y_j = k} \phi(x_j) \|\| \phi(x_j)\|\|_2 \tag{8} \]

The class probability of a query examples \( x_i \) is computed via measuring the cosine similarity between \( \phi(x_i) \) and \( \bar{c}_k \):

\[ \mathbb{P}(y = k|x_i) \propto \exp \left\{ \frac{\bar{c}_k^T \phi(x_i)}{\|\| \bar{c}_k\|\| \cdot \|\| \phi(x_i)\|\|} \right\} \tag{9} \]

To accommodate the use of cosine similarity, we use a cosine classifier[10] as our default classification model \( h(\cdot) \) when training the teacher and the student.

### 5. Experimental Setup

#### 5.1. Benchmark and Datasets

Since our problem setup is new and requires additional coarsely labeled examples during representation learning, we set up new benchmarks from three existing datasets: iNaturalist [39], TieredImageNet [29] and CIFAR100 [15]. In these new benchmarks, we ensure (via re-splitting the classes between base and novel) that every novel class has a sibling base class. We also make sure that there are at least two novel classes associated to a single coarse label to ensure that the coarse label does not automatically give away the fine label. We present the class distribution of these datasets in table 1 and some relevant information below:

<table>
<thead>
<tr>
<th>Setup</th>
<th>Base</th>
<th>Novel-Seen</th>
<th>Novel-Unseen</th>
<th>Super-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>iNat2019-CL</td>
<td>398</td>
<td>126</td>
<td>119</td>
<td>50</td>
</tr>
<tr>
<td>tieredImageNet-CL</td>
<td>498</td>
<td>60</td>
<td>50</td>
<td>34</td>
</tr>
<tr>
<td>CIFAR-100-CL</td>
<td>40</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 1. Class distribution of the benchmarks introduced in this paper.

1. **iNat2019-CL.** We construct this benchmark from the iNaturalist 2019 (iNat2019) competition dataset [39] - a fine-grained animal species classification dataset with a natural taxonomy (We use the genera level labels as supercategory). After removing species and genera with insufficient examples, we split each genus into base, novel-seen and novel-unseen.

2. **TieredImageNet-CL.** TieredImageNet [29] comes with 34 high level supercategories but different supercategories are split into base and novel in the original benchmark. To reflect the assumption that novel and base classes share coarse labels, we resplit each supercategory into base, novel-seen and novel-unseen.
3. **CIFAR-100-CL.** CIFAR-100 [15] contains 100 classes of images that can be grouped evenly into 20 supercategories. We split each supercategory into 2/2/1 for base, novel-seen and novel-unseen.

For each dataset, we split examples of each class into three buckets: 60%/20%/20%. For base classes and novel-seen classes, the 60% split is used to construct the representation set. For novel-seen and novel-unseen classes, the two 20% splits are used to form $D_{ref}$ and $D_{query}$ respectively.

### 5.2. Evaluation Protocol

We report the top-1 per class accuracy averaged across each class for all datasets (to avoid issues arising from class imbalance in iNat2019). We consider two evaluation protocols - all-way-k-shot and 5-way-k-shot classification for $k = 1, 5$. When sampling 5-way classes for evaluation, we restrict the maximum number of supercategories in each single classification task to 3 to ensure that there are at least two classes that share the same supercategory in a single task and simply identifying the supercategory alone does not yield good performance. Regardless of 5-way or all-way, we construct a classification task by sampling $k$ different reference examples from each novel class and then re-classify the performance of a model on the whole query set. The process is repeated 1000 times for all-way classification and 10,000 times for 5-way classification (following [47]) to generate statistically meaningful comparisons. In addition, we also consider the all-shot setup where we use all the examples in the 20% split for all-way classification.

### 5.3. Comparisons

To assess PAS’s representations, we establish a few representations for comparisons. These representations are trained similarly to PAS (same architecture with cosine classifiers [10]) but with different loss functions:

1. **Baseline.** Here the representation is simply obtained via training the model to classify the fine-grained examples on $D_{rep}^{fine}$.
2. **Repr-Coarse.** Similar to Repr-Fine. This representation is the feature extractor of a ConvNet trained to classify the examples from both $D_{rep}^{fine}$ and $D_{rep}^{coarse}$ into their respective supercategories.
3. **Self-training.** This representation is trained similarly to PAS except that the filtering function $g$ is removed when generating the pseudo-labels.
4. **Repr-Multi.** This multi-tasked representation is produced by training a ConvNet with two cosine classifier heads - one for classifying the fine-grained label and another for the supercategory:

$$\begin{align*}
\min_{\theta} & \quad \frac{1}{n} \sum_{(x_i, y_i) \in D_{base}} l_{CE}(f_\theta(x_i), y_i) \\
+ & \quad \frac{C}{n + n'} \sum_{(x_i, p_i) \in D_{rep}} l_{CE}(f_\theta(x_i), p_i)
\end{align*}$$

where $C$ is set to 1 for simplicity.

We also get an upper bound for representation learning techniques by training a classifier on a fully labeled dataset consisting of all training examples from both base and novel-seen classes (obtained by adding fine-grained novel class labels to $D_{rep}^{coarse}$ and combining it with $D_{rep}^{fine}$); the classifier head is discarded and the feature extractor is used for few-shot transfer as with the other methods. This feature representation is the best representation one can get from the task; hence it is an upper bound for representation learning-based FSL techniques.

When evaluating the representations, we deploy the same inference procedure (sec 4.2) for fair comparison. In addition, we compare PAS’ representation (with nearest class prototype inference) to two recent few-shot learners that deploy more sophisticated inference strategies: MetaOptNet [16] and FEAT [47] for 5-way classification. These learners are trained assuming no knowledge of the coarse labels (i.e. trained on $D_{rep}^{fine}$ without the coarse labels) since they were initially developed for the conventional few-shot learning setup. We use ResNet18 [13] as the backbone for all methods and defer training details to the supplementary materials.

### 6. Experimental Results

#### 6.1. Coarsely-labeled Data Improve FSL

We present the all-way classification result on all novel classes $C_{novel}$ in table 2 and 5-way classification result in table 3. We observe the following:

1. **Data from novel classes improves representation even without labels:** Methods that use the additional data (Repr-Multi, Self-training, PAS) outperform the Baseline that is trained only on the base classes. This is true even for Self-training, which uses only the novel class data and not the coarse labels. The performance of Self-training confirms the findings in [26].

2. **Learners trained with coarsely-labeled data outperform those trained without:** The addition of coarse label information significantly helps: Among representation learning approaches, Repr-Multi and PAS both outperform the Baseline and Self-training, which do not use the coarse label information on all-way classification; for 5-way classification, Repr-Multi slightly underperforms Self-training but PAS
<table>
<thead>
<tr>
<th>Method</th>
<th>k=1</th>
<th>5</th>
<th>all</th>
<th>k=1</th>
<th>5</th>
<th>all</th>
<th>k=1</th>
<th>5</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>iNat2019-CL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>20.46</td>
<td>39.22</td>
<td>57.22</td>
<td>28.68</td>
<td>50.68</td>
<td>67.25</td>
<td>28.14</td>
<td>50.37</td>
<td>67.49</td>
</tr>
<tr>
<td>Repr-Coarse</td>
<td>19.89</td>
<td>29.32</td>
<td>41.72</td>
<td>33.50</td>
<td>44.62</td>
<td>57.62</td>
<td>28.09</td>
<td>40.32</td>
<td>51.39</td>
</tr>
<tr>
<td>Self-training</td>
<td>22.94</td>
<td>42.17</td>
<td>59.69</td>
<td>33.18</td>
<td>54.79</td>
<td>69.85</td>
<td>29.95</td>
<td>52.11</td>
<td>69.87</td>
</tr>
<tr>
<td>Repr-Multi</td>
<td>24.72</td>
<td>41.42</td>
<td>57.34</td>
<td>38.24</td>
<td>56.77</td>
<td>70.72</td>
<td>32.03</td>
<td>51.21</td>
<td>65.88</td>
</tr>
<tr>
<td>PAS</td>
<td><strong>25.21</strong></td>
<td><strong>43.27</strong></td>
<td><strong>61.04</strong></td>
<td><strong>39.06</strong></td>
<td><strong>58.76</strong></td>
<td><strong>73.63</strong></td>
<td>30.91</td>
<td>51.85</td>
<td>69.12</td>
</tr>
<tr>
<td>Upper Bound</td>
<td>27.30</td>
<td>47.98</td>
<td>64.20</td>
<td>41.64</td>
<td>64.61</td>
<td>75.36</td>
<td>30.71</td>
<td><strong>53.77</strong></td>
<td>72.29</td>
</tr>
<tr>
<td>tieredImageNet-CL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>32.16</td>
<td>53.36</td>
<td>68.97</td>
<td>41.22</td>
<td>62.92</td>
<td>77.19</td>
<td>54.19</td>
<td>75.50</td>
<td>85.51</td>
</tr>
<tr>
<td>Repr-Coarse</td>
<td>25.69</td>
<td>37.19</td>
<td>49.76</td>
<td>38.14</td>
<td>48.83</td>
<td>62.55</td>
<td>41.64</td>
<td>55.70</td>
<td>66.32</td>
</tr>
<tr>
<td>Self-training</td>
<td>35.49</td>
<td>57.26</td>
<td>70.87</td>
<td>48.12</td>
<td>69.11</td>
<td>80.60</td>
<td><strong>54.71</strong></td>
<td><strong>75.89</strong></td>
<td><strong>86.08</strong></td>
</tr>
<tr>
<td>Repr-Multi</td>
<td>37.16</td>
<td>57.27</td>
<td>70.20</td>
<td>49.54</td>
<td>68.38</td>
<td>80.28</td>
<td>53.28</td>
<td>72.94</td>
<td>83.31</td>
</tr>
<tr>
<td>PAS</td>
<td><strong>38.11</strong></td>
<td><strong>59.08</strong></td>
<td><strong>71.84</strong></td>
<td><strong>50.60</strong></td>
<td><strong>69.52</strong></td>
<td><strong>80.40</strong></td>
<td>53.18</td>
<td>74.68</td>
<td>85.12</td>
</tr>
<tr>
<td>Upper Bound</td>
<td>42.86</td>
<td>65.68</td>
<td>76.71</td>
<td>60.03</td>
<td>80.67</td>
<td>87.14</td>
<td>55.94</td>
<td>76.96</td>
<td>86.55</td>
</tr>
<tr>
<td>CIFAR-100-CL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>20.32</td>
<td>33.24</td>
<td>42.67</td>
<td>25.50</td>
<td>39.95</td>
<td>50.45</td>
<td>34.37</td>
<td>51.80</td>
<td>64.00</td>
</tr>
<tr>
<td>Repr-Coarse</td>
<td>31.56</td>
<td>38.90</td>
<td>47.87</td>
<td>45.74</td>
<td>53.36</td>
<td>63.10</td>
<td>37.52</td>
<td>50.65</td>
<td>65.20</td>
</tr>
<tr>
<td>Self-training</td>
<td>25.68</td>
<td>42.43</td>
<td>54.93</td>
<td>32.96</td>
<td>51.42</td>
<td>63.30</td>
<td>38.24</td>
<td><strong>57.51</strong></td>
<td><strong>69.50</strong></td>
</tr>
<tr>
<td>Repr-Multi</td>
<td><strong>34.99</strong></td>
<td><strong>48.42</strong></td>
<td><strong>58.37</strong></td>
<td><strong>48.57</strong></td>
<td><strong>61.95</strong></td>
<td><strong>72.65</strong></td>
<td>37.92</td>
<td>54.91</td>
<td>65.10</td>
</tr>
<tr>
<td>PAS</td>
<td><strong>35.00</strong></td>
<td><strong>48.42</strong></td>
<td><strong>58.37</strong></td>
<td><strong>48.57</strong></td>
<td><strong>61.95</strong></td>
<td><strong>72.65</strong></td>
<td>37.92</td>
<td>54.91</td>
<td>65.10</td>
</tr>
<tr>
<td>Upper Bound</td>
<td>51.83</td>
<td>64.97</td>
<td>69.17</td>
<td>73.75</td>
<td>85.02</td>
<td>85.45</td>
<td>36.53</td>
<td>56.25</td>
<td>70.30</td>
</tr>
</tbody>
</table>

Table 2. Average top-1 per class accuracy of various representations across 1000 runs. For each novel categories, we use k=1, 5 and all reference examples. Best performing entries that leverage coarsely-labeled data are bolded. 95% confidence intervals are omitted for brevity. The full table can be found in the supplementary materials.

3. **PAS is the strongest representation overall:** On all-way-5-shot classification, PAS significantly outperforms Repr-Multi by **1.92 points** on average, and yields a **8.31 points** gain over the Baseline; On 5-way-5-shot classification, PAS outperforms Repr-Multi by **1.17 points** and the Baseline by **3.24 points**. All these results show that even though multitask-training can be used to leverage coarsely annotations, it is not as effective as PAS. In conclusion, with coarse annotations, PAS is an extremely effective way of improving FSL.

To unpack the performance gains, we also evaluate the different representations separately on the novel-seen and novel-unseen classes (Table 2). To ensure that the classification tasks are truly fine-grained, we remove supercategories that only have one child when splitting the novel classes for tieredImageNet-CL (we report the performance on novel-unseen for CIFAR-100-CL for completeness even though there is only one novel-unseen class per supercategory). We observe, as expected, that the performance...
even for completely unseen novel classes potentially help CIFAR-100 (though on TieredImageNet-CL, the gains dis-
gains from Self-training, Repr-Multi and PAS for iNat and on the novel-unseen classes we do observe performance gains are largest on the novel-seen classes. However, even stronger alignment between the prediction and ground truth.

AMI has a theoretical range of \([0, 1]\) with higher value signifying identities of examples in

\[\text{Table 4. Adjusted Mutual Information (AMI) of the predicted class and their ground truth identities.}\]

\[\text{Table 3. Average 5-way-k-shot top-1 accuracy and 95\% confidence intervals of various few-shot learners and our representations across 10000 runs. Top performing entries (excluding Upper Bound) are bolded.}\]

<table>
<thead>
<tr>
<th>Method</th>
<th>iNat2019-CL</th>
<th>tieredImageNet-CL</th>
<th>CIFAR-100-CL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(k=1)</td>
<td>5</td>
<td>(k=1)</td>
</tr>
<tr>
<td>MetaOpt</td>
<td>59.32 ± 0.22</td>
<td>72.92 ± 0.20</td>
<td>59.12 ± 0.20</td>
</tr>
<tr>
<td>FEAT</td>
<td>62.76 ± 0.22</td>
<td>76.45 ± 0.20</td>
<td>67.60 ± 0.21</td>
</tr>
<tr>
<td>Baseline</td>
<td>57.07 ± 0.20</td>
<td>73.68 ± 0.19</td>
<td>65.17 ± 0.20</td>
</tr>
<tr>
<td>Repr-Coarse</td>
<td>54.43 ± 0.19</td>
<td>65.29 ± 0.18</td>
<td>56.92 ± 0.19</td>
</tr>
<tr>
<td>Self-training</td>
<td>60.19 ± 0.22</td>
<td>75.82 ± 0.20</td>
<td>68.35 ± 0.21</td>
</tr>
<tr>
<td>Repr-Multi</td>
<td>59.06 ± 0.20</td>
<td>73.74 ± 0.19</td>
<td>66.51 ± 0.20</td>
</tr>
<tr>
<td>PAS</td>
<td>59.74 ± 0.21</td>
<td>74.88 ± 0.20</td>
<td>68.02 ± 0.20</td>
</tr>
<tr>
<td>Upper Bound</td>
<td>62.64 ± 0.22</td>
<td>78.52 ± 0.19</td>
<td>73.03 ± 0.20</td>
</tr>
</tbody>
</table>

6.1.1 The Effect of Filtering

From table 2, we observe that PAS significantly outperforms Self-training. As reported in [26], the key to good transferrability of self-trained student representation relies on the alignment between the grouping induced by the teacher and the ground truth of the additional data. We posit that filtering has strengthened the alignment and thus yields a superior result. To validate this, we investigate the generated pseudo-labels on \(D_{rep}^{coarse}\) by the teacher before and after filtering. Specifically, we use the most probable prediction of the pseudo-label to “label” each example in \(D_{rep}^{coarse}\). Then, as in [26], we evaluate the induced grouping by measuring the adjusted mutual information (AMI) [40] between the induced clustering and the ground truth. Table 4 shows that the AMI increases significantly with filtering, indicating a stronger alignment between the grouping induced by the filtered pseudo-labels and the ground truth class distinctions as compared to the original pseudo-labels. We believe that this alignment results in a cleaner signal for training the student’s representation.

6.2. Analyses

In this section, we analyze PAS on iNat2019-CL. Unless explicitly stated, we report performance on all novel classes.

6.2.1 Reducing the number of Base Classes.

Few-shot learners rely on a large diverse base dataset. The usage of coarsely-labeled data sets up the possibility of reducing this dependence. In this subsection, we investigate the effect of reducing the number of base classes. To start, we remove the fine-grained labels of 2/3 of the base classes in iNat2019-CL while keeping their coarse labels. This reduces the number of base classes from 398 to only 144. As a result, the number of novel classes during evaluation (126 + 119 = 245) becomes significantly more than the effective number of classes available during representation learning (144 fine-grained, 50 coarse-grained).

We report the results on this benchmark in table 5. We find that when the number of base classes is substantially reduced, the accuracy of the baseline drops by 6 to 10 points. In contrast, we observe that with the aid of coarsely-labeled data, PAS experiences a much smaller performance degradation. PAS is thus less reliant on the availability of large amounts of fine-grained labels on the base dataset.

6.2.2 Effect of Unseen Supercategories

So far we have assumed that each novel class shares a coarse label with a base class. However, it is crucial that few-shot learners generalize to completely unseen parts of the class taxonomy. To test how PAS works in this setting, we constructed another modification of iNat2019-CL: we randomly chose a fifth of the coarse categories and removed all labels associated with these supercategories in the representation set. In particular, we removed all base classes that

\[9058\]
Large Reduction in Base Classes

<table>
<thead>
<tr>
<th>Method</th>
<th>k=1</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline - Original</td>
<td>20.46</td>
<td>39.22</td>
</tr>
<tr>
<td>PAS - Original</td>
<td>25.31</td>
<td>43.27</td>
</tr>
<tr>
<td>Baseline</td>
<td>14.19 (↓ 6.26)</td>
<td>28.37 (↓ 10.84)</td>
</tr>
<tr>
<td>PAS</td>
<td>21.80 (↓ 3.41)</td>
<td>37.31 (↓ 5.96)</td>
</tr>
</tbody>
</table>

Table 5. Average k-shot performance of different representations evaluated on the original iNat2019-CL novel classes. PAS - Original and Baseline - Original are trained on the original base dataset. (↓) indicates absolute amount of degradation due to reduced base classes. 95% confidence intervals can be found in the supplementary materials. See section 6.2.1 for more details.

Removing Some Coarse Labels

<table>
<thead>
<tr>
<th>Method</th>
<th>k=1</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>18.15</td>
<td>35.72</td>
</tr>
<tr>
<td>Repr-Multi</td>
<td>21.92 (+ 3.77)</td>
<td>37.23 (+ 1.50)</td>
</tr>
<tr>
<td>PAS</td>
<td>23.14 (+ 4.99)</td>
<td>40.73 (+ 5.00)</td>
</tr>
</tbody>
</table>

Table 6. Average k-shot performance (on iNat2019-CL novel classes) of various representations trained on a base dataset with unseen supercategories. (+) indicates absolute improvement from the Baseline. 95% confidence intervals can be found in the supplementary materials. See section 6.2.2 for details.

belong to this set of supercategories, and we removed the coarse labels corresponding to these supercategories from the coarsely-labeled dataset. Note that in this setup, the connection between base and novel classes is weakened which could impact the performance of PAS.

We adapt PAS to this setting by simply using this newly-unlabeled data with unfiltered pseudo-labels when training the student representation. In table 6, we observe that with this modification, PAS is still able to leverage all the available data to yield considerable performance gains. PAS outperforms both the baseline as well as Repr-Multi, indicating the efficacy of PAS in leveraging additional data that are less related to the base dataset.

6.2.3 Effect of Coarser Labels

Even though the distinction at the genus rank for iNat2019 is rather clear-cut (as shown in figure 1), one can also easily obtain coarser labels that correspond to higher taxonomic rank. For instance, one can recognize bees and butterflies as insects which corresponds to the kingdom of the species; one might also label bees and wasps as insects that have transparent wings (Hymenoptera) which corresponds to the order of the species.

In this section, we investigate how using these coarser labels would affect PAS. We conjecture that the coarser labels would dilute the effect of filtering and thus leading to degradation in performance. To investigate, we look into three taxonomic ranks available in iNat2019 (in decreasing order of coarseness): Kingdom (5 supercategories), Order (27 supercategories) and Genus (50 supercategories). Indeed, we find that the coarser labels are less effective, though we still see considerable gains from the “Order” level coarse labels (Figure 3). These gains roughly correlate with the AMI of the predicted class labels induced by the filtered pseudo-labels (with different coarse labels) and the ground truth.

Additional Analyses: We show the following additional results in the supplementary: (a) PAS can bring more improvements when coupled with a semi-supervised inference approach when the coarse labels of the reference set is available, and (b) the strength of PAS’ representation is correlated with the amounts of coarsely-labeled data. For more details, please see the supplementary materials.

7. Conclusion

We investigate the use of coarsely-labeled data in building more transferrable representations for few-shot learning. We found that representations that are built using the additional coarsely-labeled examples are significantly better than their counterparts when evaluated under 1-shot and 5-shot classification in three different datasets. We develop a new representation learning technique - PAS that leverages self-training and parent consistent filtering to achieve stronger representations, bringing forth enormous improvement to few-shot learning.

Acknowledgements: This work was funded by the DARPA Learning with Less Labels program (HR001118S0044).
References


[22] Lu Liu, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. Many-class few-shot learning on multi-granularity class hierarchy. IEEE Transactions on Knowledge and Data Engineering, 2020. 2


9060
[38] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In Proceedings of the European Conference on Computer Vision (ECCV), 2020. 1, 2