Webly Supervised Fine-Grained Recognition: Benchmark Datasets and An Approach

Zeren Sun[1], Yazhou Yao[1]*, Xiu-Shen Wei[1,2]*, Yongshun Zhang[2],
Fumin Shen[3], Jianxin Wu[2], Jian Zhang[4], Heng Tao Shen[3]
[1]School of Computer Science and Engineering, Nanjing University of Science and Technology
[2]State Key Laboratory for Novel Software Technology, Nanjing University
[3]University of Electronic Science and Technology of China [4]University of Technology Sydney

Abstract

Learning from the web can ease the extreme dependence of deep learning on large-scale manually labeled datasets. Especially for fine-grained recognition, which targets at distinguishing subordinate categories, it will significantly reduce the labeling costs by leveraging free web data. Despite its significant practical and research value, the webly supervised fine-grained recognition problem is not extensively studied in the computer vision community, largely due to the lack of high-quality datasets. To fill this gap, in this paper we construct two new benchmark webly supervised fine-grained datasets, termed WebFG-496 and WebiNat-5089, respectively. In concretely, WebFG-496 consists of three sub-datasets containing a total of 53,339 web training images with 200 species of birds (Web-bird), 100 types of aircrafts (Web-aircraft), and 196 models of cars (Web-car). For WebiNat-5089, it contains 5089 sub-categories and more than 1.1 million web training images, which is the largest webly supervised fine-grained dataset ever. As a minor contribution, we also propose a novel webly supervised method (termed "Peer-learning") for benchmarking these datasets. Comprehensive experimental results and analyses on two new benchmark datasets demonstrate that the proposed method achieves superior performance over the competing baseline models and states-of-the-art. Our benchmark datasets and the source codes of Peer-learning have been made available at https://github.com/NUST-Machine-Intelligence-Laboratory/weblyFG-dataset.

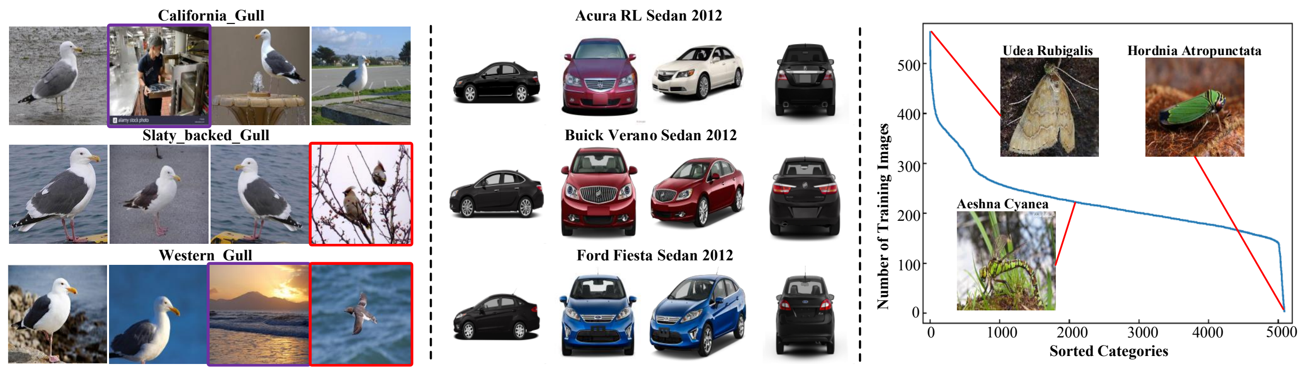
1. Introduction

Recent success of deep learning has shown that a deep network in conjunction with abundant well-labeled training data is the most promising approach for fine-grained recognition [1, 2, 19, 20]. However, even with the availability of scalable crowd-sourcing platforms like Amazon Mechanical Turk, constructing a large-scale fine-grained dataset like iNat2017 [4] is still an extremely difficult work since distinguishing subtle differences among fine-grained categories (*e.g.*, different animals [5, 11], or plants [12, 17]) usually requires domain-specific expert knowledge.

To reduce the cost of manual fine-grained annotation, many methods have been proposed, which are primarily focused on semi-supervised learning [18]. These works inevitably involve various forms of human intervention and remain labor-consuming [26]. To further reduce manual annotations as well as to learn more practical fine-grained models, training directly from web images is becoming increasingly popular [13, 14, 15, 16]. Nevertheless, the lacking of a benchmark dataset makes it difficult to fairly compare the performances of these algorithms. This is the motivation of this work and we aim to provide a benchmark dataset for evaluating webly supervised fine-grained recognition algorithms.

Webly supervised fine-grained recognition consists of three challenges: (1) **Label noise** - Different from manually labeled datasets, fine-grained web images are often associated with label noises due to the influence of error-prone automatic or non-domain-expert annotations. We consider two types of label noises, which we call "cross-domain" and "cross-category" noises. To be specific, cross-domain noise is the portion of images that are not of any categories in the same fine-grained domain, *e.g.*, for birds, it is the fraction of images that do not contain a bird (cf. images with purple bounding boxes in Fig. 1 (a)). In contrast, cross-category noise is the portion of images that have the wrong labels within a fine-grained class, *e.g.*, an image of a bird with the wrong category label (cf. images with red bounding boxes in Fig. 1 (a)). According to existing works [24], deep neural networks have memorization effects, which will memorize

*Corresponding author.



(a) Label noise in WebFG-496 (b) Small inter-class variance in WebFG-496 (c) Class imbalance in WebiNat-5089

Figure 1. Examples of WebFG-496 and WebiNat-5089. (**a**) Cross-domain (purple bounding box) and cross-category (red bounding box) noises in WebFG-496. (**b**) Due to the variety of colors, poses and other factors, there are small inter-class and large intra-class variances among subcategories in the WebFG-496 dataset. (**c**) Distribution of training images per category in WebiNat-5089. The number of training images in the largest subcategory "Udea Rubigalis" is 563 while in the smallest subcategory "Hordnia Atropunctata" is only 4.

incorrectly labeled training data and cause a poor generalization performance [25]. (2) **Small inter-class variance** - As shown in Fig. 1 (b), three fine-grained sub-categories have small inter-class variance while each sub-category has large intra-class variance. Aiming at recognizing hundreds of sub-categories belonging to the same super-category is an extremely challenging task. (3) **Class imbalance** - The natural world is heavily imbalanced, as some species are much more likely to be observed [4]. As shown in Fig. 1 (c), we collected 563 web training images for "Udea Rubigalis". While for "Hordnia Atropunctata", we only gathered 4 web images for training. Extreme class imbalance is a property of the real world fine-grained categories and recognition models should can deal with it.

To address aforementioned issues, we leverage the categories in three famous fine-grained datasets, *i.e.*, FGVC-Aircraft [27], CUB200-2011 [3] and Stanford Cars [28], to construct a new webly supervised fine-grained benchmark dataset WebFG-496. To scale up the categories as well as to build a much larger and challenging fine-grained dataset, we reuse the categories in iNat2017 [4] to build a web version iNaturalist dataset, termed as WebiNat-5089. Compared to manually labeled iNat2017, WebiNat-5089 consists of 1,184,520 web training images, basically twice the number of training images (*i.e.*, 579,184) in the original iNat2017.

Furthermore, we also propose a simple yet effective learning paradigm to train robust deep fine-grained models directly from noisy web images. Our work is motivated by the following observations: **1)** The performance of deep neural networks can be boosted by learning from wrongly classified instances (*i.e.*, "hard examples") [29]. **2)** Deep neural networks always fit to "easy examples" first, and gradually adapt to "hard examples" [24]. **3)** Distinct networks have different learning abilities, and can collaboratively boost their performance by mutually communicating "useful information" [30]. Specifically, we train two deep neural networks simultaneously and let them mutually correct their classification errors. For each mini-batch of web images, each network individually feeds forward all data to separately predict the labels, based on which the input data is split into two groups G_d (instances with different predictions) and G_s (instances with identical predictions). Then, the networks update their parameters with the instances in G_d. Meanwhile, each network sorts and fetches small-loss instances in G_s as "useful knowledge", and communicates this "useful knowledge" with its peer network for correcting the errors induced by instances in G_d during updating. Extensive experiments on the new benchmark datasets demonstrate the effectiveness of our proposed approach. The main contributions of this work can be summarized as follows:

(1) We construct two webly supervised fine-grained datasets, *i.e.*, WebFG-496 and WebiNat-5089, for fairly benchmarking webly supervised fine-grained approaches. Specifically, WebFG-496 consists of three sub-datasets: Web-aircraft, Web-bird, and Web-car. It can help researchers promptly verify the effectiveness of their proposed methods. The large-scale WebiNat-5089 contains 5089 fine-grained subcategories and more than 1.1 million web training images. To the best of our knowledge, it is the largest webly supervised fine-grained benchmark dataset.

(2) We propose a novel deep learning paradigm, *i.e.*, Peer-learning for dealing with webly supervised fine-grained recognition. Our model jointly leverages both "hard" and "easy" examples for training, which can keep the peer networks diverged and maintain their distinct learning abilities in removing noisy images. This strategy can alleviate both the accumulated error problem in MentorNet [55] and the consensus issue in Co-teaching [30], which thus can boost the performance of webly supervised learning.

(3) We conduct extensive experiments of various baseline methods to benchmark the proposed WebFG-496 and WebiNat-5089 datasets, as well as our Peer-learning. In experiments, results on WebFG-496 and WebiNat-5089 validate the superiority of our method over states-of-the-arts, and ablation studies justify both the merits of the proposed benchmark datasets and the effectiveness of our method.

2. Related Work

2.1. Fine-Grained Recognition Datasets

In the past decade, the computer vision community has developed many fine-grained datasets covering diverse meta-categories, *e.g.*, aircrafts [27, 34], birds [3, 35], cars [28, 36, 37, 38], dogs [11, 39, 40], flowers [12], food [17], leaves [41], trees [44], insects [4], *etc.*. The statistics of popular used fine-grained datasets are provided in Table 1. Compared to coarse-grained annotations, fine-grained image labeling is an extremely difficult task and only a small number of domain experts are capable of correctly labeling them. This motivates the requirement to automatically learning fine-grained recognition models from *free* web images for discriminating large numbers of potentially visual similar categories. However, existing webly supervised datasets, like NUS-WIDE [46], WebVision [47] and OpenImages [48], are all coarse-grained datasets. Benchmarked webly supervised datasets tailored for fine-grained are required to fairly evaluate the performance of proposed approaches. Our two benchmark webly supervised fine-grained datasets, *i.e.*, WebFG-496 and WebiNat-5089, are built in this context, which can promote further studies in this learning scenario.

2.2. Webly Supervised Learning

Training fine-grained recognition models with web images usually results in poor performance due to the presence of label noises and data bias [21, 22, 23]. Statistical learning has contributed significantly to solve this problem, especially in theoretical aspects. In this work, our focus is on deep learning-based approaches. Roughly speaking, these works can be separated into four groups. The first group involves developing novel loss functions [31, 32, 33, 49, 50, 56] for dealing with label noises. The second group tries to estimate the noise transition matrix [52, 53]. The third one applies attention mechanisms to alleviate noises and data bias [54]. The last group attempts to clean the web data as a preprocessing step [29, 30, 42, 43, 55]. However, none of these works are specifically designed for fine-grained visual recognition. Our proposed Peer-learning method belongs to the last group and is proposed for webly supervised fine-grained recognition. Different from existing methods (*e.g.*, Decoupling [29] which solely uses "hard examples", MentorNet [55] and Co-teaching [30] which only explore "easy examples") our approach leverages both "easy" and "hard" examples for cross-updating two networks, can alleviate the accumulated errors and consensus issues in training webly supervised fine-grained recognition models.

Table 1. The statistics of popular fine-grained datasets. "Supervision" means the training data is manually labeled ("Manual") or collected from the web ("Web").

Domains	Dataset Name	# Train	# Classes	Supervision
Birds	CUB200-2011 [3]	5,994	200	Manual
	NABirds [5]	23,929	555	Manual
Cars	Stanford Cars [28]	8,144	196	Manual
	Census Cars [38]	512,765	2,675	Manual
Dogs	Stanford Dogs [39]	12,000	120	Manual
	Oxford Pets [40]	3,680	37	Manual
	DogSnap [11]	4,776	133	Manual
Aircraft	FGVC-Aircraft [27]	3,334	100	Manual
Flowers	Flowers 102 [12]	1,020	102	Manual
Food	VegFru [17]	29,200	292	Manual
Leaves	LeafSnap [41]	23,147	185	Manual
Trees	Urban Trees [44]	14,572	18	Manual
Natural	iNat2017 [4]	579,184	5,089	Manual
Ours	**WebFG-496**	53,339	496	Web
	WebiNat-5089	1,184,520	5,089	Web

3. Datasets Construction

In this section, we explain the construction details of our webly supervised fine-grained datasets WebFG-496 and WebiNat-5089. It should be noted that WebFG-496 consists of three sub-datasets: Web-aircraft, Web-bird, and Web-car.

Fine-Grained Categories: The first issue for building a new benchmark dataset is the fine-grained categories of web images we shall collect from the Internet. In the literature, there are three famous manually labeled fine-grained datasets, *i.e.*, FGVC-Aircraft [27], CUB200-2011 [3], and Stanford Cars [28], which contain 100 types of airplanes, 200 species of birds, and 196 categories of cars, respectively. For our WebFG-496, we follow these fine-grained datasets to reuse the category labels of them as our target fine-grained categories. Furthermore, to construct a much larger and more challenging webly supervised fine-grained benchmark dataset, we explore the 5089 categories originally presented in iNat2017 [4] to build our WebiNat-5089.

Testing Images: Since the fine-grained categories in our webly supervised datasets WebFG-496 and WebiNat-5089 come from the existing datasets as aforementioned. To save the cost of dataset construction and consider convenient comparisons with traditional fine-grained methods, we directly take the testing sets in FGVC-Aircraft, CUB200-2011, and Stanford Cars as the testing data for our WebFG-496. For WebiNat-5089, the validation set of iNat2017 is utilized as the testing data.

Web Sources: As pointed out by [51], different web sources like Google Image Search Engine (GIS), Bing Image Search Engine (BIS), Flickr, Airliners, and iNaturalist may have a significant influence on the datasets. Table 2 summarizes the web sources for testing images in our WebFG-496 and WebiNat-5089. To reduce the probability of overlap with the testing set along with to train webly supervised domain robust deep fine-grained models, we ulti-

Table 2. Detailed construction process of training data in WebFG-496 and WebiNat-5089. "Testing Source" indicates where testing images come from. "Imbalance" is the number of images in the largest class divided by the number of images in the smallest.

Dataset	Sub-dataset	Classes	Testing Source	Retrieved Images	After Broken Filtering	After Duplicated Filtering	Imbalance
WebFG-496	Web-aircraft	100	Airliners	14,817	14,772	13,503	1.2
	Web-bird	200	Flickr	29,211	29,098	18,388	1.1
	Web-car	196	GIS+ Flickr+BIS	27,959	27,895	21,448	2.3
WebiNat-5089	-	5,089	iNaturalist	1,437,483	1,434,083	1,184,520	140.8

mately choose Bing Image Search Engine (BIS) as our web source of training images.

Collecting Candidate Training Images: Top-ranking images of image search engines tend to have relatively high accuracy and lower-ranking images usually contain more and more noises. Based on the consideration of reducing noise in the collected web images, for the 496 categories in WebFG-496, we treat each category label as a query and crawl the top 150 images from the BIS. Since some special categories are much harder to photograph than others, the natural world is extremely imbalanced [4] and the realistic data obeys long-tailed distribution. For the 5089 categories in WebiNat-5089, we also treat each category label as a query but crawl as many images as possible for each query. After removing the invalid links, we obtained 71,987 images for WebFG-496 (14,817 aircraft images, 29,211 bird images, and 27,959 car images) and 1,437,483 images for WebiNat-5089, respectively.

Removing Broken Images: Since our training images are directly crawled from the web, some broken images may be included. To remove these broken images, we employ the Python library `Pillow` to check each collected image and subsequently convert them to the RGB mode. Images that cannot be opened by `Pillow` or cannot be converted to RGB will be regarded as broken ones and get deleted. Table 2 gives a detailed number of images for WebFG-496 and WebiNat-5089 after broken images removing.

Removing Duplicated Images: To remove duplicated images between training data and testing data, we take advantage of deep Convolutional Neural Network (CNN) in semantic information extraction. Our overlap removing strategy is under the assumption that images with more similar semantic information are more likely to be similar or even identical. Specifically, we first use the VGG-16 [57] model pre-trained on ImageNet to extract the embedding feature vector for each image in both training and testing data. Here we select the feature maps of the last max-pooling layer and then perform global average pooling to cast them into a 512-d feature vector. Then, for every single test image per category, we calculate the similarity distance between this testing image and every training image. For each category, we obtain the smallest distance between training and testing data, which is denoted as θ. We set an empirical threshold factor η to scale the distance to $(1+\eta)\times\theta$ and remove the web training images which have a smaller distance than $(1+\eta)\times\theta$. In dataset constructions, the value of η is set to be 0.01. As shown in Table 2, after duplicated images removing, we ultimately obtain 53,339 training images for WebFG-496, and 1,184,520 training images for WebiNat-5089. After that, we manually check the obtained images according to the ranked distance between training and test data, and few duplicated images are left.

Class Imbalance in WebiNat-5089: WebiNat-5089 contains 2,101 types of plants, 1,021 kinds of insects, 289 species of reptiles, *etc*. The average number of images per class for the WebiNat-5089 dataset is 232.7 while the median number is 221. The images for some categories are easy to obtain (*e.g.*, 563 images for "Udea Rubigalis") while others are hard (*e.g.*, 4 images for "Hordnia Atropunctata"), making the extreme class imbalance a property (*i.e.*, long-tailed) for our WebiNat-5089 dataset. Therefore, training fine-grained models from web images also needs to take the class imbalance problem into account.

Dataset Accuracy: It is difficult to manually establish the accuracy of fine-grained web data, especially for WebiNat-5089 which contains over 1.1 million web training images. However, we can roughly estimate the accuracy of training data for WebFG-496 and WebiNat-5089 by random sampling. For WebFG-496, we randomly select 100 sub-categories and 50 images for each sub-category. For WebiNat-5089, we randomly select 200 sub-categories and 50 images for each sub-category. Finally, the roughly estimated accuracy of training data is shown in Table 3.

Table 3. Rough accuracy of training data estimated by random sampling for WebFG-496 and WebiNat-5089.

Dataset	WebiNat-5089	WebFG-496		
		Web-aircraft	Web-bird	Web-car
Accuracy	36%	73%	65%	67%

4. Proposed Peer-learning Network

Training strategy As shown in Fig. 2, our framework consists of two networks h_1 and h_2, which mutually communicate useful information for boosting the final performance. Specifically, suppose we have a mini-batch of data $G = \{(x_i, y_i)\}$, where y_i is the label with noise of the image x_i. h_1 and h_2 first separately predict the labels $\{\hat{y}_{i,h_1}\}$ and $\{\hat{y}_{i,h_2}\}$ of $\{x_i\}$, based on which G is divided into $G_s = \{(x_k, y_k) \in G | \hat{y}_{k,h_1} = \hat{y}_{k,h_2}\}$ (instances with identical predictions) and $G_d = \{(x_k, y_k) \in G | \hat{y}_{k,h_1} \neq \hat{y}_{k,h_2}\}$ (instances with different predictions). Inspired by [29], we treat G_d as "hard examples", which can benefit the train-

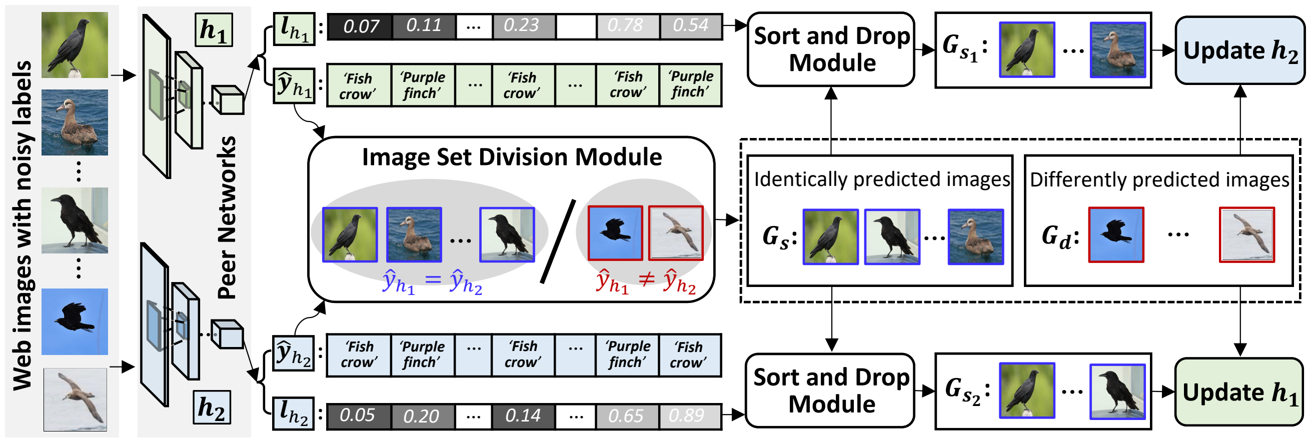

Web images with noisy labels
Peer Networks
h_1
h_2
l_{h_1}: 0.07 0.11 ... 0.23 0.78 0.54
$\hat{y}_{h_1}$: 'Fish crow' 'Purple finch' ... 'Fish crow' ... 'Fish crow' 'Purple finch'
Image Set Division Module
$\hat{y}_{h_1} = \hat{y}_{h_2}$
$\hat{y}_{h_1} \neq \hat{y}_{h_2}$
$\hat{y}_{h_2}$: 'Fish crow' 'Purple finch' ... 'Fish crow' ... 'Purple finch' 'Fish crow'
l_{h_2}: 0.05 0.20 ... 0.14 ... 0.65 0.89
Sort and Drop Module
G_{s_1}:
Update h_2
Identically predicted images
G_s:
Differently predicted images
G_d:
Sort and Drop Module
G_{s_2}:
Update h_1

Figure 2. Architecture of our Peer-learning model. The input is a mini-batch of web images. Each network in h_1 and h_2 individually feeds forward data to separately predict the labels, based on which the input data is split into two sets G_s (instances with identical predictions) and G_d (instances with different predictions). Then, h_1 and h_2 individually sort and fetch small-loss instances in G_s as the useful knowledge G_{s_1} and G_{s_2}. Subsequently, h_1 updates its parameters using G_d and G_{s_2}, while h_2 updates its parameters using G_d and G_{s_1}.

ing of h_1 and h_2. To alleviate the negative effects caused by label noises in G_d, we explore "useful knowledge" in G_s by selecting a small proportion of instances G_{s_1} and G_{s_2}, according to the losses computed by h_1 and h_2, respectively. G_{s_i} $(i \in \{1,2\})$ consists of instances that have the top $(1-d(T))$ smallest training losses by using h_i, and is concretely defined as:

$$G_{s_i} = \arg\min_{\tilde{G}_s \subset G_s : |\tilde{G}_s| \geq (1-d(T))|G_s|} \sum_{(x_j,y_j)\in\tilde{G}_s} \mathcal{L}_{h_i}(x_j,y_j), \tag{1}$$

where $\mathcal{L}_{h_i}(x_j,y_j)$ is the training loss of instance x_j computed by h_i $(i \in \{1,2\})$, and $|G_s|$ indicates the number of elements in G_s. Particularly,

$$d(T) = \xi \cdot \min\{\tfrac{T}{T_k}, 1\}, \tag{2}$$

is the drop rate that dynamically controls $|G_{s_i}|$, where ξ is the maximum drop rate, and T_k is the number of epochs after which $d(T)$ is no longer updated. The motivation lies in that we attempt to utilize more images for training at the beginning by setting $d(T)$ to a small value. Thereafter, $d(T)$ gradually increases, so that we only select correctly labeled instances with sufficiently high confidence, and drop noisy ones before the networks memorize them.

After obtaining G_d and G_{s_i} $(i \in \{1,2\})$, we treat $G_d \cup G_{s_1}$ as the "useful knowledge" to train h_2, provided by its peer network h_1. Similarly, $G_d \cup G_{s_2}$ is adopted to train h_1. The parameters θ_{h_i} of the network h_i $(i \in \{1,2\})$ are updated by using the gradient $\nabla\mathcal{L}_{h_i}$ with a learning rate λ:

$$\theta_{h_1} \leftarrow \theta_{h_1} - \lambda \cdot \sum_{(x_i,y_i)\in G_d\cup G_{s_2}} \nabla\mathcal{L}_{h_1}(x_i,y_i), \tag{3}$$

$$\theta_{h_2} \leftarrow \theta_{h_2} - \lambda \cdot \sum_{(x_i,y_i)\in G_d\cup G_{s_1}} \nabla\mathcal{L}_{h_2}(x_i,y_i). \tag{4}$$

Either network in our method learns "useful knowledge" from its peer network, which can correct errors caused by instances in G_d with label noises. Therefore, we denote our algorithm as "Peer-learning". Through mutually communicating "useful knowledge", both the performance of h_1 and h_2 can be improved.

Fine-grained module It is worth noting that the training strategy is independent of the backbone network structure h_1 and h_2. In the Peer-learning network, we realize fine-grained recognition by using existing fine-grained modules (*e.g.*, B-CNN [59], NTS-Net [6], H-B-Pooling [7], DFL-CNN [8], OPAM [9], and Fast-MPN-COV [10]) as the backbone network structure for h_1 and h_2.

Why is our approach effective? Since instances in G_d include biased predictions (at least one network gives a wrong prediction), updating the parameters only using instances in G_d may result in errors and these errors will be directly transferred back to itself in the next mini-batch. These errors will be increasingly accumulated and previous works, like Decoupling [29] and MentorNet [55], cannot handle them explicitly. In our approach, as two networks have different learning abilities, they can communicate "useful knowledge" (instances with a small loss in G_s) with their peer network to update the parameters. Through these exchange procedures, errors induced by updating with G_d can be identified and reduced by peer networks mutually. This is the reason why our approach is more robust to previous works like Decoupling and MentorNet. On the other side, compared to methods that update the model using only "clean" samples (*e.g.*, Co-teaching [30]), the advantage of our approach is that we can also leverage the so-called "noisy" samples to promote model optimization. The motivation is that, in the group G_d, there are still many correctly classified instances. Thus, we can still learn "useful knowledge" from these instances to improve the model performance as long as the label noise is conquered.

Table 4. The comparison of classification accuracy (%) for benchmark methods and webly supervised baselines (Decoupling, Co-teaching, and our Peer-learning) on the WebFG-496 dataset.

Type	Method	Backbone	WebFG-496			
			Web-Bird	Web-Aircraft	Web-Car	Average
Benchmarks	VGG-16 [57]	-	66.34	68.38	61.62	65.45
	VGG-19 [57]	-	67.69	70.99	67.21	68.63
	ResNet-50 [60]	-	64.43	60.79	60.64	61.95
	ResNet-101 [60]	-	66.74	63.46	65.51	65.24
	GoogLeNet [61]	-	66.01	66.02	65.87	65.97
	B-CNN [59]	VGG-16	66.56	64.33	67.42	66.10
Webly	Decoupling [29]	B-CNN	70.56	**75.97**	75.00	73.84
	Co-teaching [30]	B-CNN	73.85	72.76	73.10	73.24
	Peer-learning	B-CNN	**76.48**	74.38	**78.52**	**76.46**

Table 5. The comparison of classification accuracy (%) for Peer-learning by using different fine-grained backbone modules on the WebFG-496 dataset.

Method	Backbone	WebFG-496
Peer-learning	NTS-Net [6]	78.17
	H-B-Pooling [7]	77.06
	DFL-CNN [8]	76.03
	OPAM [9]	75.32
	Fast-MPN-COV [10]	77.10

Table 6. The comparison of classification accuracy (%) of benchmarks and our proposed webly supervised baseline Peer-learning on the WebiNat-5089 dataset.

Type	Method	Backbone	WebiNat-5089
Benchmarks	VGG-16 [57]	-	44.77
	GoogLeNet [61]	-	39.71
	ResNet-50 [60]	-	48.23
Webly	**Peer-learning**	ResNet-50	**54.56**

Why do we need to update the neural networks with G_d? For each instance in G_d, at least one network in h_1 and h_2 has been given the wrong predictions. These wrong predictions give rise to an updated model that can promisingly result in better classification performance. However, updating neural networks h_1 and h_2 with G_d also brings some errors. For example, the labels of instance x predicted by h_1 and h_2 are $+1$ and -1, respectively. If the ground truth label of instance x is $+1$, then h_2 treats x as a wrong prediction and updating the parameters will promote the robustness of h_2. At the same time, h_1 correctly predicts the label of instance x, but wrongly regards it has been given an error prediction and so updates its parameters by treating x as a wrong prediction will bring errors to h_1. What's worse, these errors will be transferred back and increasingly accumulated.

Why can "useful knowledge" correct the errors induced by G_d? Memorization effects [24] indicate that, on noisy data sets, deep neural networks will first learn the clean and easy patterns in the initial epochs. Thus, they can filter out noisy instances through their loss values at the beginning of training. However, as the number of epochs increases, deep neural networks will eventually overfit on noisy labels. Our key idea is to drop these noisy labels before they are memorized. In our approach, we dynamically control the drop rate of noisy labels by the parameter $d(T)$. As the number of epoch T increases, we gradually increase the drop rate $d(T)$, so that we can keep "clean" instances and drop noisy labels before the neural networks memorize them. Intuitively, small-loss instances in G_s (easy examples) are more likely to be that correctly classified. On the other hand, networks h_1 and h_2 have different decision boundaries and thus have different abilities to learn. When training on noisy web images, they can have different abilities to filter out noisy labels. We cross-update neural networks using "useful knowledge" from each other. Through these exchange procedures of useful knowledge, different types of errors induced by G_d can be mutually identified and reduced by peer networks. This process is similar to "peer-review". When students check their own papers, it is very hard for them to find errors because they have some personal bias for the answers. Fortunately, they can ask peer classmates to review their papers.

5. Experiments

5.1. Benchmarking Two Proposed Datasets

Experimental Setups: For the benchmark methods on WebFG-496, we choose Bilinear-CNN (B-CNN) [59] and five deep neural networks, including VGG-16 [57], VGG-19 [57], ResNet-50 [60], ResNet-101 [60], and GoogLeNet [61]. All the networks are pre-trained on ImageNet and then fine-tuned on three sub-datasets of WebFG-496. Specifically, we follow [59] and adopt a two-stage training strategy. We firstly freeze the convolutional layer parameters and only optimize the last fully connected layers. Then, we optimize the parameters of all layers in the fine-tuned model. In experiments, we use SGD as optimizer. Learning rate and batch size in the first stage are set to 0.01 and 64, while in the second stage, they are 0.001

Table 7. The comparison of classification accuracy (%) for using (or not) web images as the data augmentation. Improvement over the baseline model is reported as (Δ).

Testing data	**Training data**	**Backbone**	**ACA (%)**	**Improvement**
FGVC-Aircraft	FGVC-Aircraft	VGG-16	84.8	Δ **3.6**
	Web-aircraft + FGVC-Aircraft		88.4	
CUB200-2011	CUB200-2011	VGG-16	77.7	Δ **8.0**
	Web-bird + CUB200-2011		85.7	
Stanford Cars	Stanford Cars	VGG-16	85.6	Δ **6.8**
	Web-car + Stanford Cars		92.4	

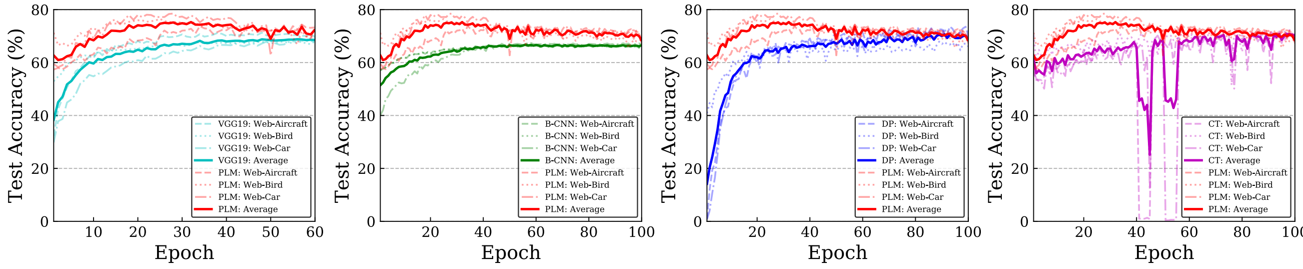


Figure 3. Comparisons of classification accuracy (%) among our Peer-learning model (PLM), VGG-19, B-CNN, Decoupling (DP), and Co-teaching (CT) on sub-datasets Web-aircraft, Web-bird, and Web-car in WebFG-496. The value on each sub-dataset is plotted in the dotted line and the average value is plotted in solid line. Note that the classification accuracy is the result of the second stage in the two-stage training strategy. Since we have trained 60 epochs in the second stage on VGG-19, we only compare the first 60 epochs in the second stage of our approach with VGG-19.

and 64. As WebiNat-5089 contains more than 1.1 million training images, it takes huge training time on such a large-scale dataset. To be efficient, for the benchmarks on WebiNat-5089, we conduct three deep networks, *i.e.*, VGG-16 [57], GoogLeNet [61], and ResNet-50 [60]. Similar to WebFG-496, we also fine-tune these deep neural networks on WebiNat-5089 but adopt a single-stage training strategy. Except for the batch size is different (1024 for WebiNat-5089), other parameters are the same.

Quantitative Results: Experimental results for benchmarks on WebFG-496 and WebiNat-5089 are presented in Table 4 and Table 6. As shown in Table 4, we can notice that all six networks achieve reasonable fine-grained classification accuracy, which validates the reliability of WebFG-496. From Table 6, we can find that the performance of three networks on WebiNat-5089 is obviously below that on WebFG-496. It indicates that WebiNat-5089 is more challenging than WebFG-496, which might be caused by the extreme class imbalance problem.

Data Augmentation: The widely-used FGVC benchmark datasets (*e.g.*, CUB200-2011) all suffer from limited training data, which severely prevented the FGVC task from being sufficiently benefited from the high learning capability of deep CNNs. As a new dataset for fine-grained recognition, it is important to prove that webly supervised data can benefit fully-supervised data. To this end, we follow the semi-supervised manner and leverage web images (*i.e.*, Web-aircraft, Web-bird, Web-car) as data augmentation w.r.t. the labeled training data for training deep FGVC models. Specifically, by pre-training VGG-16 with our webly fine-grained datasets, the improvements over baselines without leveraging webly data are shown in Table 7.

Label Smoothing: To investigate the impact of using label smoothing on webly supervised benchmark datasets, we conduct experiments on VGG-16, VGG-19, RestNet-50 and ResNet-101 by setting the smoothing parameter to be 0.1. The experimental results on WebFG-496 are given in Fig. 4. From that figure, we can observe that: 1) The performances of VGG-16 and VGG-19 do not change too much after using label smoothing on three sub-datasets. 2) When using label smoothing, the performance for ResNet-50 decreases on Web-aircraft and Web-bird, but increases obviously on Web-car. 3) The performance of ResNet-101 grows incontestably on all sub-datasets after using label smoothing.

5.2. Peer-learning Performance

Implementation Details: For WebFG-496, we use a pre-trained VGG-16 [57] to initialize all convolutional layers of both h_1 and h_2. Network architecture of h_1 and h_2 is B-CNN [59]. To avoid learning two identical networks, we perform “Kaiming Normal Initialization” over the fully connected layers to ensure that h_1 and h_2 have different starting points. In experiments, we set $\xi = 0.35$ and $T_k = 10$ as the default values on WebFG-496. For model training, we optimize by Adam and also adopt a two-stage training strategy. Learning rate and batch size in the first stage are set to 10^{-3} and 64, while in the second stage, they are 10^{-4} and 32. For WebiNat-5089, we take pre-trained

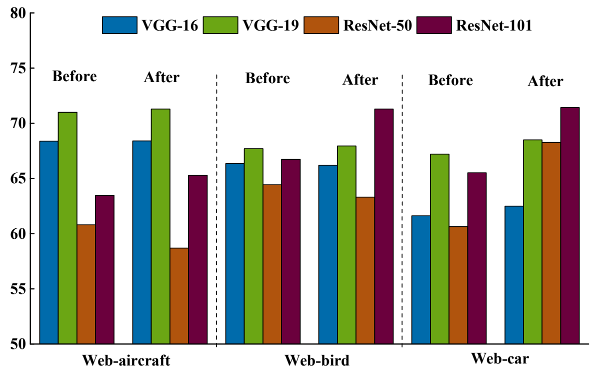


Figure 4. Comparisons of classification accuracy (%) between using ("After") label smoothing and without using ("Before") label smoothing on WebFG-496.

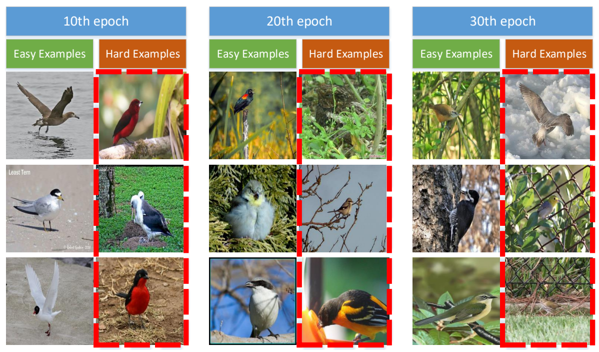


Figure 5. Visualization of "easy" and "hard" examples explored by our proposed approach as optimization continues in different training epochs.

ResNet-50 [60] to initialize h_1 and h_2. Network architecture of h_1 and h_2 is also ResNet-50. We set $\xi = 0.3$ and $T_k = 10$ on WebiNat-5089. For model training, a single-stage training strategy is performed to save training time. Learning rate, batch size, and training epoch are set to 10^{-3}, 160, and 60, respectively.

Baselines: On WebFG-496, we compare our Peer-learning with two webly supervised state-of-the-art baseline approaches: Decoupling [29] and Co-teaching [30]. To be fair, we replace the basic network in these two methods and use the same backbone network B-CNN [59] as ours. For all other parameters like batch sizes, learning rates, weight decay, training epochs, and drop rates, we all set the same as our Peer-learning. In addition, we replace the backbone network of B-CNN [59] in Peer-learning and use SOTA fine-grained modules NTS-Net [6], H-B-Pooling [7], DFL-CNN [8], OPAM [9], and Fast-MPN-COV [10] for comparison. On WebiNat-5089, we directly compare Peer-learning with benchmarks. The reason is that training a deep model on such a large-scale dataset needs more than 240 hours under the computing configuration: 4 V100 GPU (32G) cards with a batch size 160.

Quantitative Results: Experimental results are shown in Table 4, Table 5, Table 6, and Fig. 3. As shown in Table 4, we can notice that our Peer-learning greatly improves the performances of webly supervised methods. Compared to Co-teaching, our Peer-learning can not only leverage the useful knowledge in G_{s1} and G_{s2} to update the network parameters, but can also use the noisy data in G_d for promoting the parameter optimization. Compared to Decoupling, each network in our approach can learn useful knowledge from its peer network to correct the errors when updating. Thus, the errors increasingly accumulated in Decoupling can be greatly reduced in our approach. From Table 4 and Table 6, we can observe that our approach performs much better than baseline networks. Compared to [57, 60, 61], our approach maintains two networks that can learn useful knowledge from each other. From Table 5, we can notice that our Peer-learning can combine with different SOTA fine-grained modules and achieve leading results. By observing Fig. 3, we can find that our approach is not only better than other methods, but also much faster to achieve model optimization. The explanation is that our approach can gradually drop the noisy images and select useful samples for training models.

Visualization of Different Epochs: Fig. 5 visualizes "easy" and "hard" examples explored by our method in the 10-*th*, 20-*th* and 30-*th* epoch, respectively. From that figure, we can notice that with the grows of training epochs, the recognition ability of our model is increasing. For example, compared to the 10-*th* epoch, our model in the 30-*th* epoch can recognize more complex examples.

6. Conclusion

In this work, we studied the problem of fine-grained recognition through noisy web images. To be specific, we first constructed two benchmark webly supervised fine-grained datasets WebFG-496 and WebiNat-5089. Then we proposed a novel method, termed Peer-learning, by simultaneously training two deep neural networks, both of which exploit and mutually communicate useful knowledge from noisy web images. Extensive experiments showed that our approach achieved state-of-the-art performance, compared to existing works. In the future, we plan to investigate the impact of class imbalance and design a more robust fine-grained model that takes class imbalance into account.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61976116 and 61772256), the Fundamental Research Funds for the Central Universities (No. 30920021135 and 30920041111), and the CAAI-Huawei MindSpore Open Fund (CAAIXSJLJJ-2020-022A).

References

[1] Xiu-Shen Wei, Chen-Wei Xie, Jianxin Wu, and Chunhua Shen. Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recognition*, 76:704–714, 2018. 1

[2] Weifeng Ge, Xiangru Lin, and Yizhou Yu. Weakly supervised complementary parts models for fine-grained image classification from the bottom up. In *CVPR*, pages 3034–3043, 2019. 1

[3] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. *CNS-TR-2011-001*, pages 1–8, 2011. 2, 3

[4] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *CVPR*, pages 8769–8778, 2018. 1, 2, 3, 4

[5] Grant Van Horn, Steve Branson, Ryan Farrell, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition APP and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *CVPR*, pages 595–604, 2015. 1, 3

[6] Ze Yang, Tiange Luo, Dong Wang, Zhiqiang Hu, Jun Gao, and Liwei Wang. Learning to navigate for fine-grained classification. In *ECCV*, pages 420–435, 2018. 5, 6, 8

[7] Chaojian Yu, Xinyi Zhao, Qi Zheng, Peng Zhang, Xinge You. Hierarchical bilinear pooling for fine-grained visual recognition. In *ECCV*, pages 574–589, 2018. 5, 6, 8

[8] Yaming Wang, Vlad I Morariu, and Larry S Davis. Learning a discriminative filter bank within a CNN for fine-grained recognition. In *CVPR*, pages 4148–4157, 2018. 5, 6, 8

[9] Yuxin Peng, Xiangteng He, Junjie Zhao. Object-Part Attention Model for Fine-grained Image Classification. In *TIP*, 2018. 5, 6, 8

[10] Peihua Li, Jiangtao Xie, Qilong Wang, and Zilin Gao. Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In *CVPR*, pages 947–955, 2018. 5, 6, 8

[11] Jiongxin Liu, Angjoo Kanazawa, David Jacobs, and Peter Belhumeur. Dog breed classification using part localization. In *ECCV*, pages 172–185, 2012. 1, 3

[12] M-E Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *CVPR*, pages 1447–1454, 2006. 1, 3

[13] Chuanyi Zhang, Yazhou Yao, Xing Xu, Jie Shao, Jingkuan Song, Zechao Li, and Zhenmin Tang, "Extracting useful knowledge form noisy web images via data purification for fine-grained recognition," In *ACM MM*, 2021. 1

[14] Yazhou Yao, Zeren Sun, Chuanyi Zhang, Fumin Shen, Qi Wu, Jian Zhang, and Zhenmin Tang, "Jo-src: A contrastive approach for combating noisy labels," In *CVPR*, pages 5192–5201, 2021. 1

[15] Huafeng Liu, Chuanyi Zhang, Yazhou Yao, Xiushen Wei, Fumin Shen, Zhenmin Tang, and Jian Zhang, "Exploiting web images for fine-grained visual recognition by eliminating open-set noise and utilizing hard examples," *TMM*, 2021. 1

[16] Yazhou Yao, Xiansheng Hua, Guanyu Gao, Zeren Sun, Zhibin Li, and Jian Zhang, "Bridging the web data and fine-grained visual recognition via alleviating label noise and domain mismatch," In *ACM MM*, pages 1735–1744, 2020. 1

[17] Saihui Hou, Yushan Feng, and Zilei Wang. VegFru: A domain-specific dataset for fine-grained visual categorization. In *ICCV*, pages 541–549, 2017. 1, 3

[18] Yin Cui, Feng Zhou, Yuanqing Lin, and Serge Belongie. Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In *CVPR*, pages 1153–1162, 2016. 1

[19] Guo-Sen Xie, Li Liu, Xiao-Bo Jin, Fan Zhu, Zheng Zhang, Jie Qin, Yazhou Yao, and Ling Shao Attentive Region Embedding Network for Zero-shot Learning. In *CVPR*, pages 9384–9393, 2019. 1

[20] Guosen Xie, Li Liu, Fan Zhu, Fang Zhao, Zheng Zhang, Yazhou Yao, Jie Qin, Ling Shao. Region Graph Embedding Network for Zero-Shot Learning. In *ECCV*, pages 562–580, 2020. 1

[21] Zeren Sun, Xian-Sheng Hua, Yazhou Yao, Xiu-Shen Wei, Guosheng Hu, and Jian Zhang, "Crssc: salvage reusable samples from noisy data for robust learning," In *ACM MM*, pages 92–101, 2020. 3

[22] Chuanyi Zhang, Yazhou Yao, Xiangbo Shu, Zechao Li, Zhenmin Tang, and Qi Wu, "Data-driven meta-set based fine-grained visual recognition," In *ACM MM*, pages 2372–2381, 2020. 3

[23] Yazhou Yao, Fumin Shen, Guosen Xie, Li Liu, Fan Zhu, Jian Zhang, and Heng Tao Shen, "Exploiting web images for multi-output classification: From category to subcategories," *TNNLS*, 31(7): 2348–2360, 2020. 3

[24] Devansh Arpit, Stanisaw Jastrzbski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *ICML*, pages 233–242, 2017. 1, 2, 6

[25] Xinlei Chen and Abhinav Gupta. Webly supervised learning of convolutional networks. In *ICCV*, pages 1431–1439, 2015. 2

[26] Yazhou Yao, Tao Chen, Guo-Sen Xie, Chuanyi Zhang, Fumin Shen, Qi Wu, Zhenmin Tang, and Jian Zhang, "Non-salient region object mining for weakly supervised semantic segmentation," In *CVPR*, 2623–2632, 2021. 1

[27] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, pages 1–6, 2013. 2, 3

[28] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, pages 554–561, 2013. 2, 3

[29] Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". In *NIPS*, pages 960–970, 2017. 2, 3, 4, 5, 6, 8

[30] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NIPS*, pages 8527–8537, 2018. 2, 3, 5, 6, 8

[31] Yazhou Yao, Xian-sheng Hua, Fumin Shen, Jian Zhang, and Zhenmin Tang, “A domain robust approach for image dataset construction,” In *ACM MM*, pages 212–216, 2016. 3

[32] Yazhou Yao, Fumin Shen, Jian Zhang, Li Liu, Zhenmin Tang, and Ling Shao, “Extracting multiple visual senses for web learning,” *TMM*, 21(1): 184–196, 2019. 3

[33] Yazhou Yao, Fumin Shen, Jian Zhang, Li Liu, Zhenmin Tang, and Ling Shao, “Extracting privileged information for enhancing classifier learning,” *TIP*, 28(1): 436–450, 2019. 3

[34] Andrea Vedaldi, Siddharth Mahendran, Stavros Tsogkas, Subhransu Maji, Ross Girshick, Juho Kannala, Esa Rahtu, Iasonas Kokkinos, Matthew B Blaschko, David Weiss, et al. Understanding objects in detail with fine-grained attributes. In *CVPR*, pages 3622–3629, 2014. 3

[35] Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *CVPR*, pages 2011–2018, 2014. 3

[36] Yen-Liang Lin, Vlad I Morariu, Winston Hsu, and Larry S Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *ECCV*, pages 466–480, 2014. 3

[37] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, pages 3973–3981, 2015. 3

[38] Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. Fine-grained car detection for visual census estimation. In *AAAI*, pages 4502–4508. 3

[39] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *CVPR Workshops*, pages 1–2, 2011. 3

[40] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, pages 3498–3505, 2012. 3

[41] Neeraj Kumar, Peter N Belhumeur, Arijit Biswas, David W Jacobs, W John Kress, Ida C Lopez, and João VB Soares. Leafsnap: A computer vision system for automatic plant species identification. In *ECCV*, pages 502–516, 2012. 3

[42] Chuanyi Zhang, Yazhou Yao, Huafeng Liu, Guo-Sen Xie, Xiangbo Shu, Tianfei Zhou, Zheng Zhang, Fumin Shen, and Zhenmin Tang, “Web-supervised network with softly update-drop training for fine-grained visual classification,” In *AAAI*, pages 12781–12788, 2020. 3

[43] Yazhou Yao, Jian Zhang, Fumin Shen, Xiansheng Hua, Jingsong Xu, and Zhenmin Tang, “Exploiting web images for dataset construction: A domain robust approach,” *TMM*, 19(8): 1771–1784, 2017. 3

[44] Jan D Wegner, Steven Branson, David Hall, Konrad Schindler, and Pietro Perona. Cataloging public objects using aerial and street-level images-urban trees. In *CVPR*, pages 6014–6023, 2016. 3

[45] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. In *TR-UT*, 2009.

[46] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. Nus-wide: A real-world web image database from national university of singapore. In *CIVR*, pages 368–375, 2007. 3

[47] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, pages 1–9, 2017. 3

[48] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification, 2017. 3

[49] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *ICLR*, pages 1–11, 2015. 3

[50] David Flatow and Daniel Penner. On the robustness of convnets to training on noisy labels. Technical report, Stanford University, 2017. 3

[51] Yazhou Yao, Jian Zhang, Fumin Shen, Li Liu, Fan Zhu, Dongxiang Zhang, and Heng Tao Shen. Towards automatic construction of diverse, high-quality image datasets. *TKDE*, 32(6): 1199–1211, 2020. 3

[52] Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *ICLR*, pages 1–9, 2017. 3

[53] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, pages 1944–1952, 2017. 3

[54] Bohan Zhuang, Lingqiao Liu, Yao Li, Chunhua Shen, and Ian Reid. Attend in groups: a weakly-supervised deep learning framework for learning from web data. In *CVPR*, pages 1878–1887, 2017. 3

[55] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, pages 1–20, 2017. 2, 3, 5

[56] Kun Yi and Jianxin Wu. Probabilistic End-to-end Noise Correction for Learning with Noisy Labels. In *CVPR*, pages 7017–7025, 2019. 3

[57] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, pages 1–14, 2015. 4, 6, 7, 8

[58] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *ACL*, pages 92–100, 1998.

[59] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *ICCV*, pages 1449–1457, 2015. 5, 6, 7, 8

[60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6, 7, 8

[61] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 6, 7, 8