Rethinking Class-Balanced Methods for Long-Tailed Visual Recognition from a Domain Adaptation Perspective

Muhammad Abdullah Jamal[†*] Matthew Brown[♯] Ming-Hsuan Yang[‡♯] Liqiang Wang[†] Boqing Gong[♯]
[†]University of Central Florida   [‡]University of California at Merced   [♯]Google

Abstract

Object frequency in the real world often follows a power law, leading to a mismatch between datasets with long-tailed class distributions seen by a machine learning model and our expectation of the model to perform well on all classes. We analyze this mismatch from a domain adaptation point of view. First of all, we connect existing class-balanced methods for long-tailed classification to target shift, a well-studied scenario in domain adaptation. The connection reveals that these methods implicitly assume that the training data and test data share the same class-conditioned distribution, which does not hold in general and especially for the tail classes. While a head class could contain abundant and diverse training examples that well represent the expected data at inference time, the tail classes are often short of representative training data. To this end, we propose to augment the classic class-balanced learning by explicitly estimating the differences between the class-conditioned distributions with a meta-learning approach. We validate our approach with six benchmark datasets and three loss functions.

1. Introduction

Big curated datasets, deep learning, and unprecedented computing power are often referred to as the three pillars of recent advances in visual recognition [32, 44, 37]. As we continue to build the big-dataset pillar, however, the power law emerges as an inevitable challenge. Object frequency in the real world often exhibits a long-tailed distribution where a small number of classes dominate, such as plants and animals [51, 1], landmarks around the globe [41], and common and uncommon objects in contexts [35, 23].

In this paper, we propose to investigate long-tailed visual recognition from a domain adaptation point of view. The long-tail challenge is essentially a mismatch problem between datasets with long-tailed class distributions seen by a machine learning model and our expectation of the

---
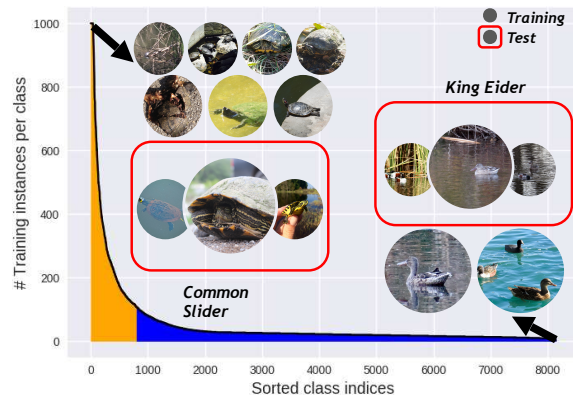[*]Work done while M. Jamal was an intern at Google.

Figure 1. The training set of iNaturalist 2018 exhibits a long-tailed class distribution [1]. We connect domain adaptation with the mismatch between the long-tailed training set and our expectation of the trained classifier to perform equally well in all classes. We also view the prevalent class-balanced methods in long-tailed classification as the target shift in domain adaptation, i.e., $P_s(y) \neq P_t(y)$ and $P_s(x|y) = P_t(x|y)$, where P_s and P_t are respectively the distributions of the source domain and the target domain, and x and y respectively stand for the input and output of a classifier. We contend that the second part of the target shift assumption does not hold for tail classes, e.g., $P_s(x|King\ Eider) \neq P_t(x|King\ Eider)$, because the limited training images of *King Eider* cannot well represent the data at inference time.

model to perform well on all classes (and not bias toward the head classes). Conventional visual recognition methods, for instance, training neural networks by a cross-entropy loss, overly fit the dominant classes and fail in the underrepresented tail classes as they implicitly assume that the test sets are drawn i.i.d. from the same underlying distribution as the long-tailed training set. Domain adaptation explicitly breaks the assumption [46, 45, 21]. It discloses the inference-time data or distribution (*target domain*) to the machine learning models when they learn from the training data (*source domain*).

Denote by $P_s(x, y)$ and $P_t(x, y)$ the distributions of a source domain and a target domain, respectively, where x and y are respectively an instance and its class label. In long-tailed visual recognition, the marginal class distribution $P_s(y)$ of the source domain is long-tailed, and yet the

class distribution $P_t(y)$ of the target domain is more balanced, e.g., a uniform distribution.

In generic domain adaptation, there could be multiple causes of mismatches between two domains. Covariate shift [46] causes domain discrepancy on the marginal distribution of input, i.e., $P_s(x) \neq P_t(x)$, but often maintains the same predictive function across the domains, i.e., $P_s(y|x) = P_t(y|x)$. Under the target-shift cause [58], the domains differ only by the class distributions, i.e., $P_s(y) \neq P_t(y)$ and $P_s(x|y) = P_t(x|y)$, partially explaining the rationale of designing class-balanced weights to tackle the long-tail challenge [7, 38, 13, 62, 40, 8, 3, 12, 39, 14].

These class-balanced methods enable the tail classes to play a bigger role than their sizes suggest in determining the model's decision boundaries. The class-wise weights are inversely related to the class sizes [26, 40, 38]. Alternatively, one can derive these weights from the cost of misclassifying an example of one class to another [13, 62]. Cui et al. proposed an interesting weighting scheme by counting the "effective number" of examples per class [7]. Finally, over/under-sampling head/tail classes [8, 39, 3, 12, 14] effectively belongs to the same family as the class-balanced weights, although they lead to practically different training algorithms. Section 2 reviews other methods for coping with the long-tail challenge.

One the one hand, the plethora of works reviewed above indicate that the target shift, i.e., $P_s(y) \neq P_t(y)$ and $P_s(x|y) = P_t(x|y)$, is generally a reasonable assumption based on which one can design effective algorithms for learning unbiased models from a training set with a long-tailed class distribution. On the other hand, however, our intuition challenges the second part of the target shift assumption; in other words, $P_s(x|y) = P_t(x|y)$ may not hold. While a head class (e.g., *Dog*) of the training set could contain abundant and diverse examples that well represent the expected data at inference time, the tail classes (e.g., *King Eider*) are often short of representative training examples. As a result, training examples drawn from the conditional distribution $P_s(x|Dog)$ of the source domain can probably well approximate the conditional distribution $P_t(x|Dog)$ of the target domain, but the discrepancy between the conditional distributions $P_s(x|King\ Eider)$ and $P_t(x|King\ Eider)$ of the two domains is likely big because it is hard to collect training examples for *King Eider* (cf. Figure 1).

To this end, we propose to augment the class-balanced learning by relaxing the assumption that the source and target domains share the same conditional distributions $P_s(x|y)$ and $P_t(x|y)$. By explicitly accounting for the differences between them, we arrive at a two-component weight for each training example. The first part is inherited from the classic class-wise weighting, carrying on its effectiveness in various applications. The second part corresponds to the conditional distributions, and we estimate it

by the meta-learning framework of learning to re-weight examples [43]. We make two critical improvements over this framework. One is that we can initialize the weights close to the optima because we have substantial prior knowledge about the two-component weights as a result of our analysis of the long-tailed problem. The other is that we remove two constraints from the framework such that the search space is big enough to cover the optima with a bigger chance.

We conduct extensive experiments on several datasets, including both long-tailed CIFAR [31], ImageNet [10], and Places-2 [61], which are artificially made long-tailed [7, 36], and iNaturalist 2017 and 2018 [51, 1], which are long-tailed by nature. We test our approach with three different losses (cross-entropy, focal loss [34], and a label-distribution-aware margin loss [4]). Results validate that our two-component weighting is advantageous over the class-balanced methods.

2. Related work

Our work is closely related to the class-balanced methods briefly reviewed in Section 1. In this section, we discuss domain adaptation and the works of other types for tackling the long-tailed visual recognition.

Metric learning, hinge loss, and head-to-tail knowledge transfer. Hinge loss and metric learning are flexible tools for one to handle the long-tailed problem [4, 26, 57, 59, 24, 54]. They mostly contain two major steps. One is to sample or group the data being aware of the long-tailed property, and the other is to construct large-margin losses. Our approach is loss-agnostic, and we show it can benefit different loss functions in the experiments. Another line of research is to transfer knowledge from the head classes to the tail. Yin et al. transfer intra-class variance from the head to tail [56], Liu et al. add a memory module to the neural networks to transfer semantic features [36], and Wang et al. employ a meta network to regress network weights between different classes [53].

Hard example mining and weighting. Hard example mining is prevalent and effective in object detection [14, 39, 44, 34]. While it is not particularly designed for the long-tailed recognition, it can indirectly shift the model's focus to the tail classes, from which the hard examples usually originate (cf. [7, 11, 16] and our experiments). Nonetheless, such methods could be sensitive to outliers or unnecessarily allow a minority of examples to dominate the training. The recently proposed instance weighting by meta-learning methods [43, 47] alleviate those issues. Following the general meta-learning principle [15, 28, 33], they set aside a validation set to guide how to weigh the training examples by gradient descent. Similar schemes are used in learning from noisy data [29, 9, 52].

Domain adaptation. In real-world applications, there often exist mismatches between the distributions of training data and test data for various reasons [49, 17, 60]. Domain adaptation methods aim to mitigate the mismatches so that the learned models can generalize well to the inference-time data [46, 45, 21, 20]. There are some approaches that handle the imbalance problem in domain adaptation. Zou et al. [63] deal with the class imbalance by controlling the pseudo-label learning and generation using the confidence scores that are normalized class-wise. Yan et al. [55] use a weighted maximum mean discrepancy to handle the class imbalance in unsupervised domain adaptation. We understand the long-tail challenge in visual recognition from the perspective of domain adaptation. While domain adaptation methods need to access a large amount of unlabeled (and sometimes also a small portion of labeled) target domain data, we do not access any inference-time data in our approach. Unlike existing weighting methods in domain adaptation [5, 27, 58], we meta-learn the weights.

3. Class balancing as domain adaptation

In this section, we present a detailed analysis of the class-balanced methods [26, 38, 7, 8, 39] for long-tailed visual recognition from the domain adaptation point of view.

Suppose we have a training set (source domain) $\{(x_i, y_i)\}_{i=1}^n$ drawn i.i.d. from a long-tailed distribution $P_s(x, y)$ — more precisely, the marginal distribution $P_s(y)$ of classes are heavy-tailed because, in visual recognition, it is often difficult to collect examples for rare classes. Nonetheless, we expect to learn a visual recognition model to make as few mistakes as possible on all classes:

$$\texttt{error} = \mathbb{E}_{P_t(x,y)} L(f(x; \theta), y), \tag{1}$$

where we desire a target domain $P_t(x, y)$ whose marginal class distribution $P_t(y)$ is more balanced (e.g., a uniform distribution) at the inference time, $f(\cdot; \theta)$ is the recognition model parameterized by θ, and $L(\cdot, \cdot)$ is a 0-1 loss. We abuse the notation $L(\cdot, \cdot)$ a little and let it be a differentiable surrogate loss (i.e., cross-entropy) during training.

Next, we apply the importance sampling trick to connect the expected $\texttt{error}$ with the long-tailed source domain,

$$\texttt{error} = \mathbb{E}_{P_t(x,y)} L(f(x; \theta), y) \tag{2}$$

$$= \mathbb{E}_{P_s(x,y)} L(f(x; \theta), y) P_t(x, y) / P_s(x, y) \tag{3}$$

$$= \mathbb{E}_{P_s(x,y)} L(f(x; \theta), y) \frac{P_t(y) P_t(x|y)}{P_s(y) P_s(x|y)} \tag{4}$$

$$:= \mathbb{E}_{P_s(x,y)} L(f(x; \theta), y) w_y (1 + \tilde{\epsilon}_{x,y}), \tag{5}$$

where $w_y = P_t(y)/P_s(y)$ and $\tilde{\epsilon}_{x,y} = P_t(x|y)/P_s(x|y) - 1$.

Existing class-balanced methods focus on how to determine the class-wise weights $\{w_y\}$ and result in the follow-

ing objective function for training,

$$\min_\theta \quad \frac{1}{n} \sum_{i=1}^n w_{y_i} L(f(x_i; \theta), y_i), \tag{6}$$

which approximates the expected inference $\texttt{error}$ (eq. (5)) by assuming $\tilde{\epsilon}_{x,y} = 0$ or, in other words, by assuming $P_s(x|y) = P_t(x|y)$ for any class y. This assumption is referred to as target shift [58] in domain adaptation.

We contend that the assumption of a shared conditional distribution, $P_s(x|y) = P_t(x|y)$, does not hold in general, especially for the tail classes. One may easily compile a representative training set for *Dog*, but not for *King Eider*. We propose to explicitly model the difference $\tilde{\epsilon}_{x,y}$ between the source and target conditional distributions and arrive at an improved algorithm upon the class-balanced methods.

4. Modeling the conditional differences

For simplicity, we introduce a conditional weight $\epsilon_{x,y} := w_y \tilde{\epsilon}_{x,y}$ and re-write the expected inference $\texttt{error}$ as

$$\texttt{error} = \mathbb{E}_{P_s(x,y)} L(f(x; \theta), y)(w_y + \epsilon_{x,y}) \tag{7}$$

$$\approx \frac{1}{n} \sum_{i=1}^n (w_{y_i} + \epsilon_i) L(f(x_i; \theta), y_i), \tag{8}$$

where the last term is an unbiased estimation of the $\texttt{error}$. Notably, we do not make the assumption that the conditional distributions of the source and target domains are the same, i.e., we allow $P_s(x|y) \neq P_t(x|y)$ and $\epsilon_i \neq 0$. Hence, the weight for each training example consists of two parts. One component is the class-wise weight w_{y_i}, and the other is the conditional weight ϵ_i. We need to estimate both components to derive a practical algorithm from eq. (8) because the underlying distributions of data are unknown — although we believe the class distribution of the training set must be long-tailed.

4.1. Estimating the class-wise weights $\{w_y\}$

We let the class-wise weights resemble the empirically successful design in the literature. In particular, we estimate them by the recently proposed "effective numbers" [7]. Supposing there are n_y training examples for the y-th class, we have $w_y \approx (1 - \beta)/(1 - \beta^{n_y})$ where $\beta \in [0, 1)$ is a hyper-parameter with the recommended value $\beta = (n - 1)/n$, and n is the number of training examples.

4.2. Meta-learning the conditional weights $\{\epsilon_i\}$

We estimate the conditional weights by customizing a meta-learning framework [43]. We describe our approach below and then discuss two critical differences from the original framework in Section 4.3.

The main idea is to hold out a balanced development set D from the training set and use it to guide the search for

the conditional weights that give rise to the best-performing recognition model $f(\cdot; \theta)$ on the development set. Denote by T the remaining training data. We seek the conditional weights $\boldsymbol{\epsilon} := \{\epsilon_i\}$ by solving the following problem,

$$\min_{\boldsymbol{\epsilon}} \quad \frac{1}{|D|} \sum_{i \in D} L(f(x_i; \theta^*(\boldsymbol{\epsilon})), y_i) \text{ with} \quad (9)$$

$$\theta^*(\boldsymbol{\epsilon}) \leftarrow \arg\min_{\theta} \frac{1}{|T|} \sum_{i \in T} (w_{y_i} + \epsilon_i) L(f(x_i; \theta), y_i) \quad (10)$$

where we do *not* weigh the losses over the development set which is already balanced. Essentially, the problem above searches for the optimal conditional weights such that, after we learn a recognition model $f(\cdot; \theta)$ by minimizing the error estimation (eqs (10) and (8)), the model performs the best on the development set (eq. (9)).

It would be daunting to solve the problem above by brute-force search, e.g., iterating all the possible sets $\{\boldsymbol{\epsilon}\}$ of conditional weights. Even if we can, it is computationally prohibitive to train for each set of weights a recognition model $f(\cdot; \theta^*(\boldsymbol{\epsilon}))$ and then find out the best model from all.

Instead, we modify the meta-learning framework [43] and search for the conditional weights in a greedy manner. It interleaves the quest for the weights $\boldsymbol{\epsilon}$ with the updates to the model parameters θ, given current time step t,

$$\tilde{\theta}^{t+1}(\boldsymbol{\epsilon}^t) \leftarrow \theta^t - \eta \frac{\partial \sum_{i \in T} (w_{y_i} + \epsilon_i^t) L(f(x_i; \theta^t), y_i)}{\partial \theta}$$

$$\boldsymbol{\epsilon}^{t+1} \leftarrow \boldsymbol{\epsilon}^t - \tau \frac{\partial \sum_{i \in D} L(f(x_i; \tilde{\theta}^{t+1}(\boldsymbol{\epsilon}^t)), y_i)}{\partial \boldsymbol{\epsilon}}$$

$$\theta^{t+1} \leftarrow \theta^t - \eta \frac{\partial \sum_{i \in T} (w_{y_i} + \epsilon_i^{t+1}) L(f(x_i; \theta^t), y_i)}{\partial \theta}.$$

The first equation tries a one-step gradient descent for θ^t using the losses weighted by the current conditional weights $\boldsymbol{\epsilon}^t$ (plus the class-wise weights). The updated model parameters $\tilde{\theta}^{t+1}(\boldsymbol{\epsilon}^t)$ are then scrutinized on the balanced development set D, which updates the conditional weights by one step. The updated weights $\boldsymbol{\epsilon}^{t+1}$ are better than the old ones, meaning that the model parameters θ^{t+1} returned by the last equation should give rise to smaller recognition error on the development set than $\tilde{\theta}^{t+1}$ do. Starting from θ^{t+1} and $\boldsymbol{\epsilon}^{t+1}$, we then move on to the next round of updates. We present our overall algorithm in the next section.

4.3. Overall algorithm and discussion

We are ready to present **Algorithm** 1 for long-tailed visual recognition. The discussions in the previous sections consider all the training examples in a batch setting. **Algorithm** 1 customizes it into a stochastic setting so that we can easily integrate it with deep learning frameworks.

There are two learning stages in the algorithm. In the first stage (lines 1–5), we train the neural recognition network $f(\cdot; \theta)$ by using the conventional cross-entropy loss

---

Algorithm 1 Meta-learning for long-tailed recognition
Require: Training set T, balanced development set D
Require: Class-wise weights $\{w_y\}$ estimated by using [7]
Require: Learning rates η and τ, stopping steps t_1 and t_2
Require: Initial parameters θ of the recognition network
1: **for** $t = 1, 2, \cdots, t_1$ **do**
2:      Sample a mini-batch B from the training set T
3:      Compute loss $\mathcal{L}_B = \frac{1}{|B|} \sum_{i \in B} L(f(x_i; \theta), y_i)$
4:      Update $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_B$
5: **end for**
6: **for** $t = t_1 + 1, \cdots, t_1 + t_2$ **do**
7:      Sample a mini-batch B from the training set T
8:      Set $\epsilon_i \leftarrow 0, \forall i \in B$, and denote by $\boldsymbol{\epsilon} := \{\epsilon_i, i \in B\}$
9:      Compute $\mathcal{L}_B = \frac{1}{|B|} \sum_{i \in B} (w_{y_i} + \epsilon_i) L(f(x_i; \theta), y_i)$
10:    Update $\tilde{\theta}(\boldsymbol{\epsilon}) \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_B$
11:    Sample B_d from the balanced development set D
12:    Compute $\mathcal{L}_{B_d} = \frac{1}{|B_d|} \sum_{i \in B_d} L(f(x_i; \tilde{\theta}(\boldsymbol{\epsilon})), y_i)$
13:    Update $\boldsymbol{\epsilon} \leftarrow \boldsymbol{\epsilon} - \tau \nabla_\epsilon \mathcal{L}_{B_d}$
14:    Compute new loss with the updated $\boldsymbol{\epsilon}$
        $\tilde{\mathcal{L}}_B = \frac{1}{|B|} \sum_{i \in B} (w_{y_i} + \epsilon_i) L(f(x_i; \theta), y_i)$
15:    Update $\theta \leftarrow \theta - \eta \nabla_\theta \tilde{\mathcal{L}}_B$
16: **end for**

---

over the long-tailed training set. The second stage (lines 6–16) meta-learns the conditional weights by resorting to a balanced development set and meanwhile continues to update the recognition model. We highlight the part for updating the conditional weights in lines 11–13.

Discussion. It is worth noting some seemingly small and yet fundamental differences between our algorithm and the learning to re-weight (L2RW) method [43]. Conceptually, while we share the same meta-learning framework as L2RW, both the class-wise weight, $w_y = P_t(y)/P_s(y)$, and the conditional weight, $\epsilon_{x,y} = w_y \tilde{\epsilon}_{x,y} = P_t(y)/P_s(y)(P_t(x|y)/P_s(x|y) - 1)$, have principled interpretations as oppose to a general per-example weight in L2RW. We will explore other machine learning frameworks (e.g., [2, 48]) to learn the conditional weights in future work, but the interpretations of them remain the same.

Algorithmically, unlike L2RW, we employ two-component weights, estimate the class-wise components by a different method [7], do *not* clip negative weights $\{\epsilon_i\}$ to 0, and do *not* normalize them such that they sum to 1 within a mini-batch. The clipping and normalization operations in L2RW unexpectedly reduce the search space of the weights, and the normalization is especially troublesome as it depends on the mini-batch size. Hence, if the optimal weights actually lie outside of the reduced search space, there is no chance to hit the optima by L2RW. In contrast, our algorithm searches for each conditional weight ϵ_i in the full real space. One may wonder whether or not our total effective

weight, $w_{y_i} + \epsilon_i$, could become negative. Careful investigation reveals that it never goes below 0 in our experiments, likely due to that the good initialization (as explained below) to the conditional weights makes it unnecessary to update the weights too wildly by line 13 in **Algorithm** 1.

Computationally, we provide proper initialization to both the conditional weights, by $\epsilon_i \leftarrow 0$ (line 8), and the model parameters θ of the recognition network, by pre-training the network with a vanilla cross-entropy loss (lines 1–5). As a result, our algorithm is more stable than L2RW (cf. Section 5.1). Note that 0 is a reasonable *a priori* value for the conditional weights thanks to the promising results obtained by existing class-balanced methods. Those methods assume that the discrepancy is as small as 0 between the conditional distributions of the source and target domains, meaning that $P_t(x|y)/P_s(x|y) - 1$ is close to 0, so are the conditional weights $\{\epsilon_i\}$. Hence, our approach should perform at worst the same as the class-balanced method [7] by initializing the conditional weights to 0 (and the class-wise weights by [7]).

5. Experiments

Datasets. We evaluate and ablate our approach on six datasets of various scales, ranging from the manually created long-tailed CIFAR-10 and CIFAR-100 [7], ImageNet-LT, and Places-LT [36], to the naturally long-tailed iNaturalist 2017 [51] and 2018 [1]. Following [7], we define the *imbalance factor (IF)* of a dataset as the class size of the first head class divided by the size of the last tail class.

Long-Tailed CIFAR (CIFAR-LT): The original CIFAR-10 (CIFAR-100) dataset contains 50,000 training images and 10,000 test images of size 32x32 uniformly falling into 10 (100) classes [31]. Cui et al. [7] created long-tailed versions by randomly removing training examples. In particular, the number of examples dropped from the y-th class is $n_y \mu^y$, where n_y is the original number of training examples in the class and $\mu \in (0, 1)$. By varying μ, we arrive at six training sets, respectively, with the imbalance factors (IFs) of 200, 100, 50, 20, 10, and 1, where IF=1 corresponds to the original datasets. We do not change the test sets, which are balanced. We randomly select ten training images per class as our development set D.

ImageNet-LT: In spirit similar to the long-tailed CIFAR datasets, Liu et al. [36] introduced a long-tailed version of ImageNet-2012 [10] called ImageNet-LT. It is created by firstly sampling the class sizes from a Pareto distribution with the power value $\alpha = 6$, followed by sampling the corresponding number of images for each class. The resultant dataset has 115.8K training images in 1,000 classes, and its imbalance factor is 1280/5. The authors have also provided a validation set with

20 images per class, from which we sample ten images to construct our development set D. The original balanced ImageNet-2012 validation set is used as the test set (50 images per class).

Places-LT: Liu et al. [36] have also created a Places-LT dataset by sampling from Places-2 [61] using the same strategy as above. It contains 62.5K training images from 365 classes with an imbalance factor 4980/5. This large imbalance factor indicates that it is more challenging than ImageNet-LT. Places-LT has 20 (100) validation (test) images per class. Our development set D contains ten images per class randomly selected from the validation set.

iNaturalist (iNat) 2017 and 2018: The iNat 2017 [51] and 2018 [1] are real-world fine-grained visual recognition datasets that naturally exhibit long-tailed class distributions. iNat 2017 (2018) consists of 579,184 (435,713) training images in 5,089 (8,142) classes, and its imbalance factor is 3919/9 (1000/2). We use the official validation sets to test our approach. We select five (two) images per class from the training set of iNat 2017 (2018) for the development set.

Table 1 gives an overview of the six datasets used in the following experiments.

Evaluation Metrics. As the test sets are all balanced, we simply use the top-k error as the evaluation metric. We report results for $k = 1, 3, 5$.

5.1. Object recognition with CIFAR-LT

We run both comparison experiments and ablation studies with CIFAR-LT-10 and CIFAR-LT-100. We use ResNet-32 [25] in the experiments.

Competing methods. We compare our approach to the following competing ones.
- **Cross-entropy training.** This is the baseline that trains ResNet-32 using the vanilla cross-entropy loss.
- **Class-balanced loss [7].** It weighs the conventional losses by class-wise weights, which are estimated based on effective numbers. We apply this class-balanced weighting to three different losses: cross-entropy, the focal loss [34], and the recently proposed label-distribution-aware margin loss [4].
- **Focal loss [34].** The focal loss can be understood as a smooth version of hard example mining. It does not directly tackle the long-tailed recognition problem. However, it can penalize the examples of tail classes more than those of the head classes if the network is biased toward the head classes during training.
- **Label-distribution-aware margin loss [4].** It dynamically tunes the margins between classes according to their degrees of dominance in the training set.

Table 1. Overview of the six datasets used in our experiments. (IF stands for the imbalance factor)

Dataset	# Classes	IF	# Train. img.	Tail class size	Head class size	# Val. img.	# Test img.
CIFAR-LT-10	10	1.0–200.0	50,000–11,203	500–25	5,000	–	10,000
CIFAR-LT-100	100	1.0–200.0	50,000–9,502	500–2	500	–	10,000
iNat 2017	5,089	435.4	579,184	9	3,919	95,986	–
iNat 2018	8,142	500.0	437,513	2	1,000	24,426	–
ImageNet-LT	1,000	256.0	115,846	5	1,280	20,000	50,000
Places-LT	365	996.0	62,500	5	4,980	7,300	36,500

- **Class-balanced fine-tuning [8].** The main idea is to first train the neural network with the whole imbalanced training set and then fine-tune it on a balanced subset of the training set.
- **Learning to re-weight (L2RW) [43].** It weighs training examples by meta-learning. Please see Section 4.3 for more discussions about L2RW and our approach.
- **Meta-weight net [47].** Similarly to L2RW, it also weighs examples by a meta-learning method except that it regresses the weights by a multilayer perceptron.

Implementation details. For the first two baselines, we use the code of [7] to set the learning rates and other hyperparameters. We train the L2RW model using an initial learning rate of 1e-3. We decay the learning rate by 0.01 at the 160th and 180th epochs. For our approach, we use an initial learning rate of 0.1 and then also decay the learning rate at the 160th and 180th epochs by 0.01. The batch size is 100 for all experiments. We train all models on a single GPU using the stochastic gradient descent with momentum.

Results. Table 2 shows the classification errors of ResNet-32 on the long-tailed CIFAR-10 with different imbalance factors. We group the competing methods into three sessions according to which basic loss they use (cross-entropy, focal [34], or LDAM [4]). We test our approach with all three losses. We can see that our method outperforms the competing ones in each session by notable margins. Although the focal loss and the LDAM loss already have the capacity of mitigating the long-tailed issue, respectively, by penalizing hard examples and by distribution-aware margins, our method can further boost their performances. In general, the advantages of our approach over existing ones become more significant as the imbalance factor increases. When the dataset is balanced (the last column), our approach does not hurt the performance of vanilla losses compared to L2RW. We can draw about the same observations as above for the long-tailed CIFAR-100 from Table 3.

Where does our approach work? Figure 2 presents three confusion matrices respectively by the models of the cross-entropy training, L2RW, and our method on CIFAR-LT-10. The imbalance factor is 200. Compared with the cross-entropy model, L2RW improves the accuracies on the tail classes and yet sacrifices the accuracies for the head classes. In contrast, ours maintains about the same performance as the cross-entropy model on the head classes and

meanwhile significantly improves the accuracies for the last five tail classes.

What are the learned conditional weights? We are interested in examining the conditional weights $\{\epsilon_i\}$ for each class throughout the training. For a visualization purpose, we average them within each class. Figure 3 demonstrates how they change over the last 20 epochs for the 1st, 4th, 7th, and 10th classes of CIFAR-LT-10. The two panels correspond to the imbalance factors of 100 and 10, respectively. Interestingly, the learned conditional weights of the tail classes are more prominent than those of the head classes in most epochs. Moreover, the conditional weights of the two head classes (the 1st and 4th) are even below 0 at certain epochs. Such results verify our intuition that the scarce training examples of the tail classes deserve more attention in training to make the neural network perform in a balanced fashion at the test phase.

Ablation study: ours vs. L2RW. Our overall algorithm differs from L2RW mainly in four ways: 1) pre-training the network, 2) initializing the weights by *a priori* knowledge, 3) two-component weights, and estimating the class-wise components by a separate algorithm [7], and 4) no clipping or normalization of the weights. Table 5 examines these components by applying them one after another to L2RW. First, pre-training the neural network boosts the performance of the vanilla L2RW. Second, if we initialize the sample weights by our class-wise weights $\{w_y\}$, the errors increase a little probably because the clipping and normalization steps in L2RW require more careful initialization to the sample weights. Third, if we replace the sample weights by our two-component weights, we can bring the performance of L2RW closer to ours. Finally, after we remove the clipping and normalization, we arrive at our algorithm, which gives rise to the best results among all variations.

Ablation study: the two-component weights. By Table 5, we also highlight the importance of the two-component weights $\{w_{y_i} + \epsilon_i\}$ motivated from our domain adaptation point of view to the long-tailed visual recognition. First of all, they benefit L2RW (comparing "L2RW, pre-training, $w_{y_i} + \epsilon_i$" with "L2RW, pre-training" in Table 5). Besides, they are also vital for our approach. If we drop the class-wise weights, our results would be about the same as L2RW with pre-training. If we drop the conditional weights and meta-learn the class-wise weights (cf. "Ours

Table 2. Test top-1 errors (%) of ResNet-32 on CIFAR-LT-10 under different imbalance settings. * indicates results reported in [47].

Imbalance factor	200	100	50	20	10	1
Cross-entropy training	34.32	29.64	25.19	17.77	13.61	7.53/7.11*
Class-balanced cross-entropy loss [7]	31.11	27.63	21.95	15.64	13.23	7.53/7.11*
Class-balanced fine-tuning [8]	33.76	28.66	22.56	16.78	16.83	7.08
Class-balanced fine-tuning [8]*	33.92	28.67	22.58	13.73	13.58	**6.77**
L2RW [43]	33.75	27.77	23.55	18.65	17.88	11.60
L2RW [43]*	33.49	25.84	21.07	16.90	14.81	10.75
Meta-weight net [47]	32.8	26.43	20.9	15.55	12.45	7.19
Ours with cross-entropy loss	**29.34**	**23.59**	**19.49**	**13.54**	**11.15**	7.21
Focal loss [34]	34.71	29.62	23.29	17.24	13.34	6.97
Class-balanced focal Loss [7]	31.85	25.43	20.78	16.22	12.52	6.97
Ours with focal Loss	**25.57**	**21.1**	**17.12**	**13.9**	**11.63**	7.19
LDAM loss [4] (results reported in paper)	-	26.65	-	-	13.04	11.37
LDAM-DRW [4] (results reported in paper)	-	22.97	-	-	**11.84**	-
Ours with LDAM loss	**22.77**	**20.0**	**17.77**	**15.63**	12.6	**10.29**

Table 3. Test top-1 errors (%) of ResNet-32 on CIFAR-LT-100 under different imbalance settings. * indicates results reported in [47].

Imbalance factor	200	100	50	20	10	1
Cross-entropy training	65.16	61.68	56.15	48.86	44.29	29.50
Class-balanced cross-entropy loss [7]	64.30	61.44	55.45	48.47	42.88	29.50
Class-balanced fine-tuning [8]	61.34	58.5	53.78	47.70	42.43	29.37
Class-balanced fine-tuning [8]*	61.78	58.17	53.60	47.89	42.56	29.28
L2RW [43]	67.00	61.10	56.83	49.25	47.88	36.42
L2RW [43]*	66.62	59.77	55.56	48.36	46.27	35.89
Meta-weight net [47]	63.38	58.39	54.34	46.96	41.09	29.9
Ours with cross-entropy loss	**60.69**	**56.65**	**51.47**	**44.38**	**40.42**	**28.14**
Focal Loss [34]	64.38	61.59	55.68	48.05	44.22	**28.85**
Class-balanced focal Loss [7]	63.77	60.40	54.79	47.41	42.01	**28.85**
Ours with focal loss	**60.66**	**55.3**	**49.92**	**44.27**	**40.41**	29.15
LDAM Loss [4] (results reported in paper)	-	60.40	-	-	43.09	-
LDAM-DRW [4] (results reported in paper)	-	57.96	-	-	**41.29**	-
Ours with LDAM loss	**60.47**	**55.92**	**50.84**	**47.62**	42.0	-

Table 4. Classification errors on iNat 2017 and 2018. (*results reported in paper. CE=cross-entropy, CB=class-balanced)

Dataset	iNat 2017		iNat 2018	
Method	Top-1	Top-3/5	Top-1	Top-3/5
CE	43.49	26.60/21.00	36.20	19.40/15.85
CB CE [7]	42.59	25.92/20.60	34.69	19.22/15.83
Ours, CE	**40.62**	**23.70/18.40**	**32.45**	**18.02/13.83**
CB focal [7]*	41.92	–/20.92	38.88	–/18.97
LDAM [4]*	–	–	35.42	–/16.48
LDAM-drw*	–	–	32.00	–/14.82
cRT [30]*	–	–	34.8	–
cRT+epochs*	–	–	32.4	–

Table 5. Ablation study of our approach by using the cross-entropy loss on CIFAR-LT-10. The results are test top-1 errors%.

Imbalance factor	100	50	20
L2RW [43]	27.77	23.55	18.65
L2RW, pre-training	25.96	22.04	15.67
L2RW, pre-training, init. by w_y	26.26	22.50	17.44
L2RW, pre-training, $w_{y_i} + \epsilon_i$	24.54	20.47	14.38
Ours	**23.59**	**19.49**	**13.54**
Ours updating w_y	25.42	20.13	15.62
Class-balanced [7]	27.63	21.95	15.64

updating w_y"), the errors become larger than our original algorithm. Nonetheless, the results are better than the class-balanced training (cf. last row in Table 5), implying that the learned class-wise weights give rise to better models than the effective-number-based class-wise weights [7].

5.2. Object recognition with iNat 2017 and 2018

We use ResNet-50 [25] as the backbone network for the iNat 2017 and 2018 datasets. The networks are pre-trained on ImageNet for iNat 2017 and on ImageNet plus iNat 2017 for iNat 2018. We experiment with the mini-batch size of 64 and the learning rate of 0.01. We train all the models using the stochastic gradient descent with momentum. For the meta-learning stage of our approach, we switch to a small learning rate, 0.001.

Table 4 shows the results of our two-component weighting applied to the cross-entropy loss. We shrink the text size for iNat 2018 to signify that we advocate experiments with iNat *2017* instead because there are only three validation/test images per class in iNat 2018 (cf. Table 1). Our ap-
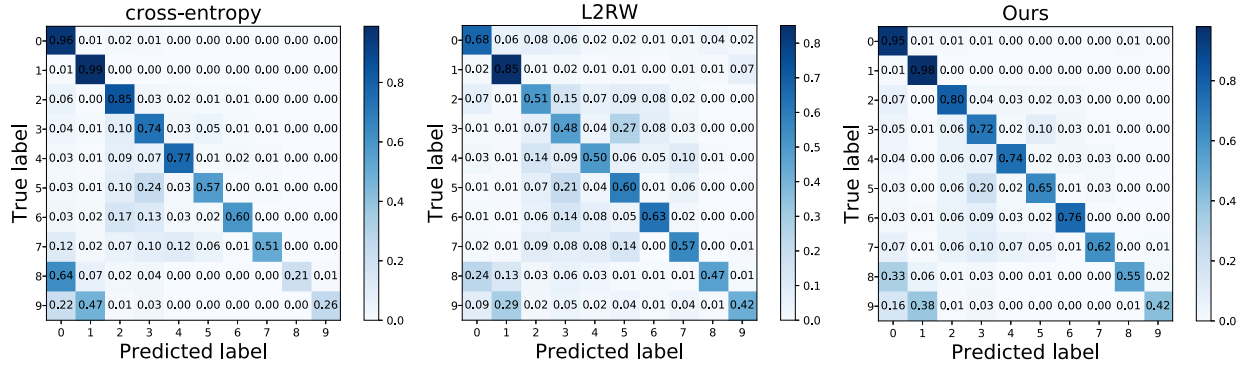
Figure 2. Confusion matrices by the cross-entropy training, L2RW, and our method on CIFAR-LT-10 (the imbalance factor is 200).
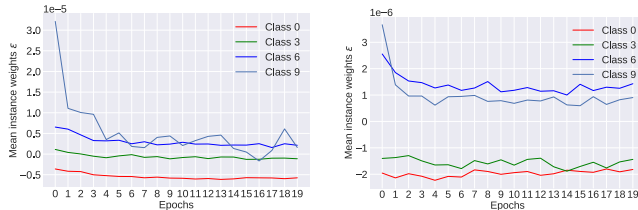
Figure 3. Mean conditional weights $\{\epsilon_i\}$ within each class vs. training epochs on CIFAR-LT-10 (left: IF = 100; right: IF = 10).

Table 6. Classification errors on ImageNet-LT and Places-LT. (*reported in paper. CE=cross-entropy, CB=class-balanced)

Dataset	ImageNet-LT		Places-LT	
Method	Top-1	Top-3/5	Top-1	Top-3/5
CE	74.74	61.35/52.12	73.00	52.05/41.44
CB CE [7]	73.41	59.22/50.49	71.14	51.58/41.96
Ours, CE	**70.10**	**53.29/45.18**	**69.20**	**47.95/38.00**

proach boosts the cross-entropy training by about 2% more than the class-balanced weighting [7] does. As we have reported similar effects for the focal loss and the LDAM loss on CIFAR-LT with extensive experiments, we do not run them on the large-scale iNat datasets to save computation costs. Nonetheless, we include the results reported in the literature of the focal loss, LDAM loss, and a classifier re-training method [30], which was published after we submitted the work to CVPR 2020.

5.3. Experiments with ImageNet-LT and Places-LT

Following Liu et al.'s experiment setup [36], we employ ResNet-32 and ResNet-152 for the experiments on ImageNet-LT and Places-LT, respectively. For ImageNet-LT, we adopt an initial learning rate of 0.1 and decay it by 0.1 after every 35 epochs. For Places-LT, the initial learning rate is 0.01 and is decayed by 0.1 every 10 epochs. For our own approach, we switch from the cross-entropy training to the meta-learning stage when the first decay of the learning rate happens. The mini-batch size is 64, and the optimizer is stochastic gradient descent with momentum.

Results. Table 6 shows that the class-balanced training improves the vanilla cross-entropy results, and our two-component weighting further boosts the results. We expect the same observation with the focal and LDAM losses. Finally, we find another improvement by updating the classification layers only in the meta-learning stage. We arrive at 62.90% top-1 error (39.86/29.87% top-3/5 error) on Places-LT, which is on par with 64.1% by OLTR [36] or 63.3% by cRT [30], while noting that our two-component weighting can be conveniently applied to both OLTR and cRT.

6. Conclusion

In this paper, we make two major contributions to the long-tailed visual recognition. One is the novel domain adaptation perspective for analyzing the mismatch problem in long-tailed classification. While the training set of real-world objects is often long-tailed with a few classes that dominate, we expect the learned classifier to perform equally well in all classes. By decomposing this mismatch into class-wise differences and the discrepancy between class-conditioned distributions, we uncover the implicit assumption behind existing class-balanced methods, that the training and test sets share the same class-conditioned distribution. Our second contribution is to relax this assumption to explicitly model the ratio between two class-conditioned distributions. Experiments on six datasets verify the effectiveness of our approach.

Future work. We shall explore other techniques [2, 48] for estimating the conditional weights. In addition to the weighting scheme, other domain adaptation methods [22, 6], such as learning domain-invariant features [18, 50] and data sampling strategies [19, 42], may also benefit the long-tailed visual recognition problem. Especially, the domain-invariant features align well with Kang et al.'s recent work on decoupling representations and classifications for long-tailed classification [30].

References

[1] iNaturalist 2018 competition dataset. https://github.com/visipedia/inat_comp/tree/master/2018, 2018. 1, 2, 5

[2] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(Sep), 2009. 4, 8

[3] Kevin W. Bowyer, Nitesh V. Chawla, Lawrence O. Hall, and William P. Kegelmeyer. SMOTE: synthetic minority oversampling technique. *arXiv:1106.1813*, 2011. 2

[4] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, pages 1565–1576, 2019. 2, 5, 6, 7

[5] Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *International conference on algorithmic learning theory*, 2008. 3

[6] Gabriela Csurka. *Domain adaptation in computer vision applications*. Springer, 2017. 8

[7] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. Class-balanced loss based on effective number of samples. *arXiv:1901.05555*, 2019. 2, 3, 4, 5, 6, 7, 8

[8] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge J. Belongie. Large scale fine-grained categorization and domain-specific transfer learning. *arXiv:1806.06193*, 2018. 2, 3, 6, 7

[9] Mostafa Dehghani, Arash Mehrjou, Stephan Gouws, Jaap Kamps, and Bernhard Schölkopf. Fidelity-weighted learning. In *ICLR*, 2018. 2

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 5

[11] Qi Dong, Shaogang Gong, and Xiatian Zhu. Class rectification hard mining for imbalanced deep learning. *arXiv:1712.03162*, 2017. 2

[12] Chris Drummond and Robert Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats oversampling. *Proceedings of the ICML'03 Workshop on Learning from Imbalanced Datasets*, 2003. 2

[13] Charles Elkan. The foundations of cost-sensitive learning. In *IJCAI*, 2001. 2

[14] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2009. 2

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

[16] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, Aug. 1997. 2

[17] Chuang Gan, Tianbao Yang, and Boqing Gong. Learning attributes equals multi-source domain generalization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 87–97, 2016. 3

[18] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv:1409.7495*, 2014. 8

[19] Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013. 8

[20] Boqing Gong, Fei Sha, and Kristen Grauman. Overcoming dataset bias: An unsupervised domain adaptation approach. In *NIPS Workshop on Large Scale Visual Recognition and Retrieval*, volume 3. Citeseer, 2012. 3

[21] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 1, 3

[22] Raghuraman Gopalan, Ruonan Li, Vishal M Patel, Rama Chellappa, et al. Domain adaptation for visual recognition. *Foundations and Trends® in Computer Graphics and Vision*, 8(4):285–378, 2015. 8

[23] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 1

[24] Munawar Hayat, Salman H. Khan, Waqas Zamir, Jianbing Shen, and Ling Shao. Max-margin class imbalanced learning with gaussian affinity. *arXiv:1901.07711*, 2019. 2

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv:1512.03385*, 2015. 5, 7

[26] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *CVPR*, 2016. 2, 3

[27] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *NeurIPS*, 2007. 3

[28] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *CVPR*, 2019. 2

[29] 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*, 2018. 2

[30] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv:1910.09217*, 2019. 7, 8

[31] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 2, 5

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 1

[33] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. Meta-sgd: Learning to learn quickly for few shot learning. *arXiv:1707.09835*, 2017. 2

[34] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *TPAMI*, 2018. 2, 5, 6, 7

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1

[36] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019. 2, 5, 8

[37] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[38] Dhruv Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. *arXiv:1805.00932*, 2018. 2, 3

[39] Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011. 2, 3

[40] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv:1310.4546*, 2013. 2

[41] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *CVPR*, 2017. 1

[42] Piyush Rai, Avishek Saha, Hal Daumé III, and Suresh Venkatasubramanian. Domain adaptation meets active learning. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, 2010. 8

[43] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv:1803.09050*, 2018. 2, 3, 4, 6, 7

[44] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2

[45] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 1, 3

[46] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000. 1, 2, 3

[47] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *NeurIPS*, 2019. 2, 6, 7

[48] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert MÃžller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 2007. 4, 8

[49] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011. 3

[50] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 8

[51] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The iNaturalist species classification and detection dataset. In *CVPR*, 2018. 1, 2, 5

[52] Yixin Wang, Alp Kucukelbir, and David M. Blei. Robust probabilistic modeling with bayesian data reweighting. In *ICML*, 2017. 2

[53] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. In *NeurIPS*, 2017. 2

[54] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *NeurIPS*, 2006. 2

[55] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *CVPR*, 2017. 3

[56] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Feature transfer learning for face recognition with under-represented data. In *CVPR*, 2019. 2

[57] Chong You, Chi Li, Daniel P. Robinson, and René Vidal. A scalable exemplar-based subspace clustering algorithm for class-imbalanced data. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *ECCV*, 2018. 2

[58] Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. In *ICML*, 2013. 2, 3

[59] Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tailed training data. In *ICCV*, pages 5419–5428, 2017. 2

[60] Yang Zhang, Philip David, Hassan Foroosh, and Boqing Gong. A curriculum domain adaptation approach to the semantic segmentation of urban scenes. *TPAMI*, 2019. 3

[61] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *TPAMI*, 2018. 2, 5

[62] Zhi-Hua Zhou and Xu-Ying Liu. On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257, 2010. 2

[63] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 3