

Pseudo-labelling meets Label Smoothing for Noisy Partial Label Learning

Darshana Saravanan¹

darshana.s@research.iiit.ac.in

Naresh Manwani¹

naresh.manwani@iiit.ac.in

Vineet Gandhi¹

vgandhi@iiit.ac.in

¹IIIT Hyderabad, India

Abstract

We motivate weakly supervised learning as an effective learning paradigm for problems where curating perfectly annotated datasets is expensive and may require domain expertise such as fine-grained classification. We focus on Partial Label Learning (PLL), a weakly-supervised learning paradigm where each training instance is paired with a set of candidate labels (partial label), one of which is the true label. Noisy PLL (NPLL) relaxes this constraint by allowing some partial labels to not contain the true label, enhancing the practicality of the problem. Our work centres on NPLL and presents a framework that initially assigns pseudo-labels to images by exploiting the noisy partial labels through a weighted nearest neighbour algorithm. These pseudo-label and image pairs are then used to train a deep neural network classifier with label smoothing. The classifier’s features and predictions are subsequently employed to refine and enhance the accuracy of pseudo-labels. We perform thorough experiments on seven datasets and compare against nine NPLL and PLL methods. We achieve state-of-the-art results in all studied settings from the prior literature, obtaining substantial gains in the simulated fine-grained benchmarks. Further, we show the promising generalisation capability of our framework in realistic, fine-grained, crowd-sourced datasets.

1. Introduction

The effectiveness of contemporary deep learning methods heavily relies on the presence of high-quality annotated data. Although this might be available for typical image classification, curating labeled datasets for fine-grained classification problems is costly, arduous and requires expert knowledge. Weakly supervised learning offers a potential solution to this challenge by learning from partially labelled or noisily labelled examples. It has been widely studied in different forms, including multi-label learning [36], semi-supervised learning [23], noisy label learning [12] etc. Our paper focuses on a weakly supervised setting called Partial Label Learning (PLL), where each training instance

is paired with a set of candidate labels (partial label), out of which one is the true label. A fundamental limitation of PLL is the assumption that the correct label is always included in the partial label. To overcome this limitation and further the practicality, Noisy PLL (NPLL), also referred to as Unreliable Partial Label Learning (UPLL) [19] was proposed. NPLL allows some partial labels to not contain the correct label. In fine-grained classification, annotators may provide multiple possible labels (e.g., multiple bird species that look similar), making NPLL techniques useful for learning from such ambiguous annotations [2].

Prior works [13, 19, 22, 26, 31] on NPLL have predominantly pursued a label disambiguation approach by maintaining and updating a distribution over label probabilities. To tackle noisy samples (samples whose correct label is not in the partial label), current methods apply a detection cum mitigation technique [13, 22, 26] which are known to cause error propagation due to the unavoidable detection errors [31]. Further, most state-of-the-art (SOTA) methods [13, 26, 31] require a two-stage training. The first stage involves warm-up training of the classifier, typically using a PLL algorithm which is followed by their proposed NPLL strategy. Determining the optimal duration for warm-up poses a challenge, since a too long or short warm-up period can negatively impact the performance [31]. Some methods also need a clean validation set to decide the warm-up duration [13, 22], which may not always be available.

In this paper, we propose a novel iterative pseudo-labelling based framework for NPLL called PALS, combining Pseudo-labelling And Label Smoothing. Unlike prior works that use the probability distribution over class labels for each image as the supervision for classifier training, PALS utilizes a pseudo-labelling strategy involving the weighted nearest neighbour algorithm, that assigns a single pseudo-label to each image. PALS builds upon the early efforts of [8], suggesting that methods featuring a strong inductive bias, such as K-Nearest Neighbors (KNN), can effectively exploit partial label information for improved disambiguation. PALS then uses a selected set of reliable image-label pairs to train the classifier. Our choice of using pseudo-labels instead of soft-label distributions

as the supervision, allows us to leverage label smoothing [24]. This imparts robustness towards the potential errors during the pseudo-labelling stage, specifically in the high noise scenario. Throughout training, we demonstrate that pseudo-labelling enhances the feature representation backbone. This, in turn, enhances the accuracy of pseudo-labelling, creating a positive feedback loop.

In a nutshell, our framework has a single trainable component (e.g. a ResNet-18 [7] model). It eliminates multi-branch networks [26] and operates seamlessly without warm-up, resulting in faster training time and lower memory requirements. We perform comprehensive experiments on seven datasets in widely different settings (varying noise rates and number of candidate labels). PALS outperforms the SOTA methods by a significant margin. In contrast to previous methods, PALS maintains its performance largely even as the noise rate increases. Performance improvements, compared to other methods, become more pronounced with increasing numbers of classes and the proportion of noise in partial labels.

In fine-grained classification, we evaluate PALS in both simulated and real-world crowd-sourced benchmarks. PALS achieves significant improvements in CUB-200, outperforming all other methods by over 5-16pp. Moving beyond the simulated PLL and NPLL benchmarks as in prior literature, we also show that PALS obtains consistent results on realistic, fine-grained, crowd-sourced datasets [21]. In line with prior works, PALS also benefits from standard regularization techniques such as Mix-up and Consistency regularization. However, PALS achieves better performance regardless of the presence or absence of regularization techniques. In summary, our work makes the following contributions:

- We propose PALS, a novel iterative pseudo-labelling based framework for NPLL, which involves KNN based pseudo-labelling and label smoothing. To our knowledge, we are the first to showcase the effectiveness of label smoothing for NPLL.
- We show notable gains over nine SOTA PLL and NPLL methods in the simulated benchmarks from prior literature. Further, we are the first to demonstrate the effectiveness of NPLL for fine-grained crowd-sourced datasets, with PALS consistently performing well across different datasets and annotator configurations.
- We present thorough ablation studies and quantitative experiments to support our design choices and demonstrate the efficacy of our approach.

2. Related Work

2.1. Traditional Partial Label Learning

Identification-based strategies (IBS) treat the ground truth as a latent variable and progressively refine the confidence

of each candidate label during training. [9] use a maximum likelihood criterion to learn the parameters of a classifier that maximizes the latent label distribution, estimated from the classifier predictions in an EM fashion. [33] propose a maximum margin formulation for PLL, which maximizes the margin between the ground-truth label and all other labels.

Average-based strategies (ABS) treat all candidate labels equally while distinguishing between the candidate and non-candidate labels. Early work by [8] extends the K-Nearest neighbours (KNN) algorithm to tackle PLL by employing majority voting among the candidate labels of neighbours. [35] also utilize KNN to classify any unseen instance based on minimum error reconstruction from its nearest neighbours. [3] maximize the average output of candidate labels and minimize the output of non-candidate labels in parametric models. PALS takes a cue from the KNN-based ABS approach [8] and uses a variation for the pseudo-labelling step.

2.2. Deep Partial Label Learning

With the advent of Deep Learning, a variety of identification-based strategies that employ a neural network backbone have been proposed. [32] temporally ensemble the predictions at different epochs to disambiguate the partial label. [16] use self-training to update the model and progressively identify the true label. [6] formulate the generation process of partial labels and develop two classifiers: one, risk-consistent and the other, classifier-consistent. [28] propose a family of loss functions that generalize the different loss functions proposed by earlier works [3, 9, 16]. [25] present PiCO, a framework that learns discriminative representations by leveraging contrastive learning and prototypical classification. [29] present a technique that uses consistency regularization on the candidate set and supervised learning on the non-candidate set. [30] upgrade PiCO and introduce self-training and prototypical alignment to achieve noteworthy results. However, none of these methods account for the possibility of noise within partial labels, which is the primary focus of our work.

2.3. Noisy Partial Label Learning

Earlier works assume that the correct label is always a part of the partial label, which limits the practicality of the problem. Hence, some recent works have diverted attention to NPLL that relaxes this condition, and allows some partial labels not to contain the correct label. [19] perform disambiguation to move incorrect labels from candidate labels to non-candidate labels and refinement to move correct labels from non-candidate labels to candidate labels. [22] separate the dataset into reliable and unreliable sets and then perform label disambiguation-based training for the former and semi-supervised learning for the latter. [13] iteratively

detect and purify the noisy partial labels, thereby reducing the noise level in the dataset. [26] extend PiCO to tackle noisy PLL by performing distance-based clean sample selection and learning a robust classifier by semi-supervised contrastive learning. [31] reduce the negative impact of detection errors by carefully aggregating the partial labels and model outputs.

Unlike the aforementioned identification-focused approaches, [17] propose Average Partial Loss (APL), a family of loss functions that achieve promising results for both PLL and NPLL. Moreover, they provide insights into how ABS can enhance IBS when used for warm-up training. Our work builds upon this intuition by alternating between ABS and IBS. Our findings demonstrate that employing K-nearest neighbours (KNN) for ABS and utilizing cross-entropy loss with label smoothing for IBS, yields SOTA performance.

3. Methodology

Figure 1 depicts the three modules of PALS that are applied sequentially in every iteration: Pseudo-labelling, Noise robust learning and Partial label augmentation. First, we assign pseudo-labels to all images using a weighted KNN. Then, for each class, we select the m most reliable image-label pairs which are used to train the classifier. To be resilient towards the potential noise in the pseudo-labelling stage, we leverage label smoothing. Finally, we augment the partial labels with the confident top-1 predictions of the classifier. The features from the updated classifier are then used for the next stage of pseudo-labelling. We hypothesize that, as training progresses, more number of samples will be assigned the correct label as the pseudo-label, which will help learn an improved classifier. Pseudo-code of PALS is available in Algorithm 1. We now provide a detailed description of each of the mentioned steps.

3.1. Problem Setup

Let \mathcal{X} be the input space, and $\mathcal{Y} = \{1, 2, \dots, C\}$ be the output label space. We consider a training dataset $\mathcal{D} = \{(\mathbf{x}_i, Y_i)\}_{i=1}^n$, where each tuple comprises of an image $\mathbf{x}_i \in \mathcal{X}$ and a partial label $Y_i \subset \mathcal{Y}$. Similar to a supervised learning setup, PLL aims to learn a classifier that predicts the correct output label. However, the training dataset in PLL contains high ambiguity in the label space, which makes the training more difficult when compared to supervised learning. A basic assumption of PLL is that the correct label y_i is always present in the partial label, i.e., $\forall i y_i \in Y_i$. In NPLL, we relax this constraint, and the correct label potentially lies outside the candidate set, i.e., $\exists i y_i \notin Y_i$.

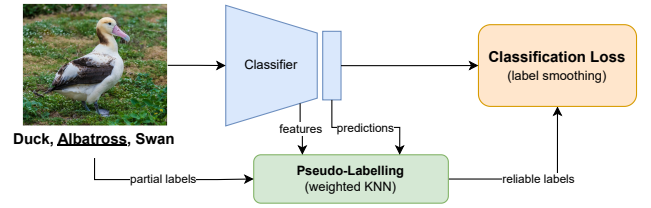


Figure 1. Illustration of PALS. Pseudo-labelling involves the usage of weighted KNN to obtain reliable image-label pairs. These pairs are used to train a classifier using label smoothing. Finally, the confident top-1 model predictions are used to augment the partial label for the upcoming iteration.

3.2. Algorithm

3.2.1. Pseudo-labelling

For each sample (\mathbf{x}_i, Y_i) , we pass the image \mathbf{x}_i through the encoder $f(\cdot)$ to obtain the feature representation \mathbf{z}_i . Then, we select its K nearest neighbours from the entire dataset. Let \mathcal{N}_i denote the set of K nearest neighbours of \mathbf{x}_i . We then compute a pseudo-label \hat{y}_i using weighted KNN as follows.

$$\hat{y}_i = \arg \max_c \sum_{\mathbf{x}_k \in \mathcal{N}_i} \mathbb{I}[c \in Y_k] \cdot s_{ik} \quad c \in [C], \quad (1)$$

where s_{ik} is the cosine similarity between \mathbf{z}_i and \mathbf{z}_k . We assume that these pseudo-labels are less ambiguous than the partial labels supposing that the samples in the neighbourhood have the same class label. Now, we approximate the class posterior probabilities \hat{q}_c via KNN, inspired by the Noisy Label Learning literature [18].

$$\hat{q}_c(\mathbf{x}_i) = \frac{1}{Z_i} \sum_{\mathbf{x}_k \in \mathcal{N}_i} \mathbb{I}[\hat{y}_k = c] \cdot s_{ik} \quad c \in [C] \quad (2)$$

where Z_i is the normalization factor such $\sum_c \hat{q}_c(\mathbf{x}_i) = 1$. Using these posterior probabilities, we select a maximum of m most reliable image-label pairs for each class where m is the δ -quantile of $[a_1, a_2, \dots, a_C]$. We do this to ensure a near-uniform distribution of samples across all classes. a_1, a_2, \dots, a_C are defined as follows.

$$a_c = \sum_{i=1}^n \mathbb{I} \left[c = \arg \max_{c'} \hat{q}_{c'}(\mathbf{x}_i) \ \& \ c \in Y_i \right], \quad c \in [C] \quad (3)$$

Thus, a_c is the number of samples for which c is the highest probable class according to the posterior probability and c is also in the partial label of those samples. For instance, when $\delta = 0$, $m = \min([a_1, a_2, \dots, a_C])$.

A key insight is that, as training progresses, the per-class agreements a_c also increase. This leads to an increase in

Algorithm 1 Pseudo-code of PALS

Input: Dataset \mathcal{D} , encoder $f(\cdot)$, classifier $h(\cdot)$, epochs T_{max} , mini-batch size B , hyper-parameters $k, \delta, \zeta, r, \lambda_{max}, \lambda_{min}$

- 1: **for** $t = 1, 2, \dots, T_{max}$ **do**
- 2: Compute psuedo-labels \hat{y}_i for all samples \mathbf{x}_i in the dataset using Eq.(1);
- 3: Compute the posterior probability vectors using Eq.(2);
- 4: Compute the maximum samples to be selected per class m using Eq.(3);
- 5: Build the Reliable image-label pairs set \mathcal{T} using Eq.(4);
- 6: Shuffle \mathcal{T} into $\frac{T}{B}$ mini-batches;
- 7: **for** $b = 1, 2, \dots, \frac{T}{B}$ **do**
- 8: Update parameters of $f(\cdot)$ and $h(\cdot)$ by minimizing loss in Eq.(10);
- 9: **end for**
- 10: Perform Partial label augmentation using Eq.(11);
- 11: **end for**

Output: parameters of $f(\cdot)$ and $h(\cdot)$

m and, consequently, the number of reliable pseudo-labels. The selection procedure of m most reliable images per class is described below.

For class c , the images for which the posterior probability approximated via KNN $\hat{q}_c(\mathbf{x}_i)$ is greater than the threshold γ_c are chosen as reliable images. Thus, the set of reliable pairs belonging to the c -th class (denoted as \mathcal{T}_c) is defined as follows.

$$\mathcal{T}_c = \{(\mathbf{x}_i, c) \mid \hat{q}_c(\mathbf{x}_i) > \gamma_c, c \in Y_i, i \in [n]\} \quad c \in [C] \quad (4)$$

For the c -th class, the threshold γ_c is dynamically defined such that a maximum of m samples can be selected per class. Note that, it is possible for a sample \mathbf{x}_i to satisfy the conditions for multiple different \mathcal{T}_c . In that case, we choose the one \mathcal{T}_c that has the highest $\hat{q}_c(\mathbf{x}_i)$. Finally, we form the reliable image-label pair set \mathcal{T} as

$$\mathcal{T} = \bigcup_{c=1}^C \mathcal{T}_c = \{(\mathbf{x}_i, \tilde{y}_i)\}_{i=1}^{\tilde{n}} \quad (5)$$

where \tilde{y}_i is the reliable pseudo-label for \mathbf{x}_i and \tilde{n} is the number of selected reliable samples.

3.2.2. Noise robust learning

We train a noise-robust classifier $h(\cdot)$ by leveraging label smoothing [24] and optional regularization components. Label smoothing uses a positively weighted average of the hard training labels and uniformly distributed soft labels to generate the smoothed labels. Let $\mathbf{e}_{\tilde{y}_i}$ be C -dimensional

one-hot encoding of label \tilde{y}_i such that its \tilde{y}_i^{th} element is one and rest of the entries are zero. The corresponding smoothed label $\mathbf{e}_{\tilde{y}_i}^{LS,r}$ is:

$$\mathbf{e}_{\tilde{y}_i}^{LS,r} = (1-r) \cdot \mathbf{e}_{\tilde{y}_i} + \frac{r}{C} \cdot \mathbf{1} \quad (6)$$

Then the per-sample classification loss is formulated as:

$$\mathcal{L}_{ce}(\mathbf{x}_i, \mathbf{e}_{\tilde{y}_i}^{LS,r}) = -(1-r) \log(h^{\tilde{y}_i}(f(\mathbf{x}_i))) - \frac{r}{C} \sum_{j=1}^C \log(h^j(f(\mathbf{x}_i))) \quad (7)$$

Although label smoothing introduces an additional hyper-parameter r , fixing it to a high value that is greater than the true noise rate ($r = 0.5$ in the experiments) improves performance [15].

3.2.3. Optional regularization components

In line with the prior works, we also enhance the performance of PALS by leveraging Mix-up [34] and Consistency regularization (CR) [1]. Consider a pair of samples $(\mathbf{x}_i, \mathbf{e}_{\tilde{y}_i}^{LS,r})$ and $(\mathbf{x}_j, \mathbf{e}_{\tilde{y}_j}^{LS,r})$. We create the Mix-up image by linearly interpolating with a factor $\alpha_{mix} \sim \text{Beta}(\zeta, \zeta)$ where ζ is a parameter in the beta distribution and define the Mix-up loss below.

$$\mathbf{x}_{ij}^{mix} = \alpha_{mix} \mathbf{x}_i + (1 - \alpha_{mix}) \mathbf{x}_j \quad (8)$$

$$\mathcal{L}_{mix}(\mathbf{x}_i, \mathbf{x}_j) = \alpha_m \mathcal{L}_{ce}(\mathbf{x}_{ij}^{mix}, \mathbf{e}_{\tilde{y}_i}^{LS,r}) + (1 - \alpha_m) \mathcal{L}_{ce}(\mathbf{x}_{ij}^{mix}, \mathbf{e}_{\tilde{y}_j}^{LS,r}) \quad (9)$$

Then, we include CR in the complete training objective. The idea is that the model should produce similar predictions for perturbed versions of the same image. We implement a variation by assigning the same pseudo-label to both weak and strong augmentations of an image.

$$\mathcal{L}_{final} = \sum_{i,j,j' \in [\tilde{n}]} \mathcal{L}_{mix}(aug_w(\mathbf{x}_i), aug_w(\mathbf{x}_j)) + \mathcal{L}_{mix}(aug_s(\mathbf{x}_i), aug_s(\mathbf{x}_{j'})) \quad i, j, j' \in [\tilde{n}] \quad (10)$$

where $aug_w(\cdot)$ and $aug_s(\cdot)$ represent weak and strong augmentation function respectively. Since Mix-up and CR are optional components, we show the effectiveness of our framework without these components in table 6.

3.2.4. Partial label augmentation

In Pseudo-labelling (Eq.4), for every sample image, we select its reliable pseudo-label only from the partial label. This limits the number of correct samples that can be selected in the NPLL case. To overcome this issue, we include the highest probability class of the current model prediction in the partial label for the next iteration if it is greater than

a threshold λ^t . λ^t is a decaying threshold and can be interpreted as balancing between precision and recall. Initially, the high threshold implies that few accurate predictions are used to avoid error propagation. The threshold is later decreased to allow more samples to be de-noised.

$$Y_i^{t+1} = \begin{cases} Y_i \cup \{\arg \max_c h_t^c(f_t(aug_w(\mathbf{x}_i)))\}, \\ \text{if } \max_c h_t^c(f_t(aug_w(\mathbf{x}_i))) > \lambda^t \\ Y_i, \text{ otherwise.} \end{cases} \quad (11)$$

Here, f_t and h_t are the encoder and classifier at the t^{th} iteration. It is important to note that, we include the highest probability class in the true partial label set Y_i to get Y_i^{t+1} . Unlike previous works [13, 31], we do not directly disambiguate the partial label. Instead, we merely include a possible candidate to allow its selection in the pseudo-labelling stage.

4. Experiments

4.1. Experimental Setup

Datasets We construct the PLL and NPLL benchmark for four different datasets: CIFAR-10, CIFAR-100, CIFAR-100H [11] and CUB-200 [27]. We follow the same dataset generation process of [26]. The generation process is governed by two parameters: the partial rate (q) and the noise rate (η). The partial rate q represents the probability of an incorrect label to be present in the candidate set and the noise rate η represents the probability of excluding the correct label from the candidate set. For CIFAR10, we experiment with $q \in \{0.1, 0.3, 0.5\}$, and for other datasets (with a larger number of classes) we select q from the set $\{0.01, 0.03, 0.05\}$. We conduct experiments using different values of $\eta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$, where values of η exceeding 0.3 indicate high noise scenarios. Further, we show the effectiveness of our approach in real-world annotation settings by considering three fine-grained crowdsourced datasets: Treeversity#6, Benthic, Plankton [21]. *Treeversity* is a publicly available dataset of plant images, crowdsourced by the Arnold Arboretum of Harvard University. *Benthic* is a collection of seafloor images containing flora and fauna. *Plankton* consists of underwater plankton images. Each image in these datasets has multiple annotations provided by domain experts.

Baselines We compare PALS against nine prior SOTA methods, four PLL and five NPLL. The PLL methods include 1) RC [6]: a risk-consistent classifier for PLL; 2) LWS [28]: employs a loss trade-off between candidate labels and non-candidate labels; 3) PiCO [25]: learns discriminative representations using contrastive and prototype learning 4) CRDPLL [29]: adapts consistency regularization for PLL. The NPLL methods include 1) PiCO+ [26]: extends PiCO for NPLL by including additional losses; 2)

FREDIS [19]: performs partial label disambiguation and refinement 3) IRNet [13]: detects and corrects the partial labels iteratively; 4) UPLLRS [22]: proposes a two-stage framework for dataset separation and disambiguation; 3) ALIM [31]: aggregates the partial labels and model outputs using two variants of normalization (Scale and OneHot). If available, we directly reference the numbers from the existing literature, and whenever possible, we present results obtained by customizing the publicly available official code repository to suit the specific configuration.

Implementation Details. To ensure a fair comparison, we use ResNet-18 [7] as the encoder $f(\cdot)$ and $h(\cdot)$ as the linear classifier. Other methods either use ResNet-18 [7] or a stronger backbone e.g. FREDIS uses ResNet32. Strong augmentation function utilizes autoaugment [4] and cutout [5]. Faiss [10] is used to speed up the nearest neighbour computation. Irrespective of the partial and noise rate, we set $K = 15$, $\delta = 0.25$, $\zeta = 1.0$, $r = 0.5$, and decrease λ_t linearly in $[0.45, 0.35]$. For CIFAR datasets, we choose the SGD optimizer with a momentum of 0.9 and set the weight decay to 0.001. The initial learning rate is set to 0.1 and decayed using a cosine scheduler, and the model is trained for 500 epochs. For CUB-200, as per the existing works, we initialize the encoder with ImageNet-1K [20] pre-trained weights. We choose the SGD optimizer with a momentum of 0.9 and set the weight decay to $5e-4$. We set the initial learning rate to 0.05 and decay it by 0.2 using the step scheduler at epochs $[60, 120, 160, 200]$ and train for 250 epochs. On crowdsourced datasets, we utilize a ResNet50 backbone pre-trained on ImageNet-1K, to align with prior efforts. We report the mean and standard deviation of the accuracy on test sets based on three runs for each experiment. All experiments are implemented with PyTorch and carried out on the NVIDIA GeForce RTX 2080 Ti GPU.

4.2. Results and Discussion

SOTA comparisons: Table 1 compares PALS with the SOTA methods on CIFAR-10 and CIFAR-100 datasets. The results are presented with three different partial rates and three different noise rates, comprising a total of nine combinations. PALS surpasses all other methods in all settings, providing clear evidence of its effectiveness. It is also apparent that the PLL approaches struggle to generalize effectively in noisy settings, highlighting the necessity for noise-resistant adaptations.

The performance of all methods drops with an increase in the size of the candidate label set (higher q) and with an increase in noise rate. A key aspect is to observe the performance across the first and last column of Table 1 (the easiest vs. the most complex setting). In several methods, the performance significantly drops; for instance, on CIFAR-10, the performance of LWS drops from 82.97 to 39.42. PALS and ALIM fare better in this regard and are able to retain

CIFAR-10	$q = 0.1$			$q = 0.3$			$q = 0.5$		
	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.3$	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.3$	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.3$
RC	80.87±0.30	78.22±0.23	75.24±0.17	79.69±0.37	75.69±0.63	71.01±0.54	72.46±1.51	59.72±0.42	49.74±0.70
LWS	82.97±0.24	79.46±0.09	74.28±0.79	80.93±0.28	76.07±0.38	69.70±0.72	70.41±2.68	58.26±0.28	39.42±3.09
FREDIS	90.57±0.23	88.01±0.19	84.35±0.20	89.02±0.15	86.14±0.17	81.02±0.60	87.42±0.21	80.81±0.99	65.15±0.13
PiCO	90.78±0.24	87.27±0.11	84.96±0.12	89.71±0.18	85.78±0.23	82.25±0.32	88.11±0.29	82.41±0.30	68.75±2.62
CRDPLL	93.48±0.17	89.13±0.39	86.19±0.48	92.73±0.19	86.96±0.21	83.40±0.14	91.10±0.07	82.30±0.46	73.78±0.55
IRNet	93.44±0.21	92.57±0.25	92.38±0.21	92.81±0.19	92.18±0.18	91.35±0.08	91.51±0.05	90.76±0.10	86.19±0.41
PiCO+	94.48±0.02	94.74±0.13	94.43±0.19	94.02±0.03	94.03±0.01	92.94±0.24	93.56±0.08	92.65±0.26	88.21±0.37
UPLLRs	95.16±0.10	-	94.65±0.23	94.32±0.21	-	93.85±0.31	92.47±0.19	-	91.55±0.38
ALIM-Scale	95.71±0.01	95.50±0.08	95.35±0.13	95.31±0.16	94.77±0.07	94.36±0.03	94.71±0.04	93.82±0.13	90.63±0.10
ALIM-Onehot	95.83±0.13	95.86±0.15	95.75±0.19	95.52±0.15	95.41±0.13	94.67±0.21	95.19±0.24	93.89±0.21	92.26±0.29
PALS	96.28±0.05	96.17±0.18	95.90±0.20	95.96±0.08	95.76±0.12	95.43±0.17	95.52±0.13	95.89±0.14	94.18±0.10

CIFAR-100	$q = 0.01$			$q = 0.03$			$q = 0.05$		
	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.3$	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.3$	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.3$
RC	52.73±1.05	48.59±1.04	45.77±0.31	52.15±0.19	48.25±0.38	43.92±0.37	46.62±0.34	45.46±0.21	40.31±0.55
LWS	56.05±0.20	50.66±0.59	45.71±0.45	53.59±0.45	48.28±0.44	42.20±0.49	45.46±0.44	39.63±0.80	33.60±0.64
FREDIS	64.73±0.28	64.53 ± 0.39	59.42±0.18	67.38±0.40	63.86±0.50	60.86±0.72	66.43±0.22	63.09 ± 0.22	56.15±0.18
PiCO	68.27±0.08	62.24±0.31	58.97±0.09	67.38±0.09	62.01±0.33	58.64±0.28	67.52±0.43	61.52±0.28	58.18±0.65
CRDPLL	68.12±0.13	65.32±0.34	62.94±0.28	67.53±0.07	64.29±0.27	61.79±0.11	67.17±0.04	64.11±0.42	61.03±0.43
IRNet	71.17±0.14	70.10±0.28	68.77±0.28	71.01±0.43	70.15±0.17	68.18±0.30	70.73±0.09	69.33±0.51	68.09±0.12
PiCO+	75.04±0.18	74.31±0.02	71.79±0.17	74.68±0.19	73.65±0.23	69.97±0.01	73.06±0.16	71.37±0.16	67.56±0.17
UPLLRs	75.73±0.41	-	71.72±0.39	-	-	-	74.73±0.24	-	70.31±0.22
ALIM-Scale	77.37±0.32	76.81±0.05	76.45±0.30	77.60±0.18	76.63±0.19	75.92±0.14	76.86±0.23	76.44±0.12	75.67±0.17
ALIM-Onehot	76.52±0.19	76.55±0.24	76.09±0.23	77.27±0.23	76.29±0.41	75.29±0.57	76.87±0.20	75.23±0.42	74.49±0.61
PALS	80.90±0.50	80.45±0.49	79.78±0.13	80.33±0.04	79.40±0.50	78.52±0.24	80.00±0.25	79.08±0.22	77.87±0.29

Table 1. Comparison of PALS with the previous state-of-the-art methods across various partial and noise rates. ‘-’ indicates missing values due to the code being unavailable.

CIFAR-100	$q = 0.01$	$q = 0.05$	$q = 0.1$
RC	75.36±0.06	74.44±0.31	73.79±0.29
LWS	64.55±1.98	50.19±0.34	44.93±1.09
PiCO	73.78±0.15	72.78±0.38	71.55±0.31
CRDPLL	79.74±0.07	78.97±0.13	78.51±0.24
PiCO+	76.29±0.42	76.17±0.18	75.55±0.21
PALS	81.46±0.13	81.00±0.26	80.77±0.12

Table 2. PLL Experiments. η is set to 0.

CIFAR-100	$q = 0.05$	$q = 0.05$
	$\eta = 0.4$	$\eta = 0.5$
FREDIS	53.44±0.39	48.05±0.20
PiCO	44.17±0.08	35.51±1.14
CRDPLL	57.10±0.24	52.10±0.36
UPLLRs	-	64.78±0.53
PiCO+	66.41±0.58	60.50±0.99
ALIM-Scale	74.98±0.16	72.26±0.25
ALIM-Onehot	71.76±0.56	69.59±0.62
PALS	76.72±0.41	74.79±0.40

Table 3. Extreme noise experiments. η values are above 0.3

their performance with increased ambiguity. Another noteworthy observation is that the performance improvement of PALS becomes more pronounced in CIFAR-100 compared to the CIFAR-10 dataset.

PLL results: While our primary focus is on NPLL, assessing performance in the noise-free setting (PLL) is crucial to ensure that the adaptations for noise do not negatively affect the zero noise performance. As most NPLL methods do not provide results in noise-free settings, we limit our comparisons primarily to the PLL methods. In Table 2, we evaluate PALS on the CIFAR-100 PLL benchmark. The results indicate that PALS also achieves superior performance in the absence of noise.

Extreme noise results: In Table 3, we present the results in extremely noisy scenarios (e.g., $\eta = 0.5$, the correct label is absent from half of the partial labels). For most methods, the performance drastically drops. For example, for PiCO+ the performance drops from 76.17 ($\eta = 0$) to 60.50 ($\eta = 0.5$). For CRDPLL, the drop is from 78.97 ($\eta = 0$) to 52.10 ($\eta = 0.5$). ALIM and PALS fare well here, with PALS obtaining the best results.

Simulated fine-grained datasets: We first evaluate PALS on the CIFAR-100H dataset, which considers label hierarchy and divides the CIFAR-100 dataset into 20 coarse labels (5 fine-grained labels per coarse label). When curating the NPLL benchmark, the candidate sets are restricted to lie within the same coarse label. PALS obtains the best results in this modified setting (Table 4).

We also evaluate on the CUB-200 bird classification dataset. Since all the images exclusively showcase birds, the candidate labels are intricately linked, presenting a heightened challenge for disambiguation. Table 4 presents the results with different q and η values. PALS outperforms other approaches by a significant margin, bringing over 7pp improvement over ALIM-Scale at $q = 0.05$ and $\eta = 0.2$.

Fine-grained crowd-sourced datasets: Prior efforts of classification on crowd-sourced datasets [21] take a data-centric approach, involving label enhancement through semi-supervised and noisy label learning, with subsequent evaluation by training a model on the enhanced dataset. Instead, we tackle this as a PLL/NPLL problem and employ PALS. As the only key change in the framework, we utilize the normalized annotation label frequency per image in the weighted KNN step. We construct the partial

Dataset	CIFAR-100H	CUB-200	CUB-200	CUB-200	CUB-200
q	0.5	0.05	0.01	0.05	0.1
η	0.2	0.2	0.0	0.0	0.0
PiCO	59.8±0.25	53.05±2.03	74.14±0.24	72.17±0.72	62.02±1.16
PiCO+	68.31±0.47	60.65±0.79	69.83±0.07	72.05±0.80	58.68±0.30
ALIM-Scale	73.42±0.18	68.38±0.47	74.44±0.68	74.26±0.32	62.88±0.90
ALIM-Onehot	72.36±0.20	63.91±0.35	-	-	-
PALS	75.16±0.59	75.76±0.27	80.71±0.21	79.46±0.25	78.73±0.36

Table 4. Simulated fine-grained datasets.

	Treeversity	Benthic	Plankton
Divide-Mix	77.84±0.52	71.72±1.21	91.79±0.51
ELR+	79.05±1.22	67.53±1.76	91.76±0.91
PiCO	84.74±0.42	51.01±0.74	24.95±0.13
PALS	81.56±0.28	77.50±0.35	90.04±0.50

(a) Ten human annotators

	Treeversity	Benthic	Plankton
Divide-Mix	76.38±1.78	71.50±0.42	91.96±0.52
ELR+	77.07±0.43	69.35±0.97	91.36±0.40
ALIM-Onehot	82.58±0.92	49.17±3.06	23.75±2.53
PALS	80.47±1.22	76.52±0.03	90.20±0.97

(b) Three human annotators

Table 5. Fine-grained crowdsourced Datasets

CIFAR-10	$q = 0.1$		$q = 0.5$	
	$\eta = 0.1$	$\eta = 0.3$	$\eta = 0.1$	$\eta = 0.3$
PiCO	90.78±0.24	84.96±0.12	88.11±0.29	68.75±2.62
PiCO+	93.64±0.19	92.18±0.38	91.07±0.02	84.08±0.42
ALIM-Scale	94.15±0.14	93.28±0.08	92.52±0.12	86.51±0.21
ALIM-Onehot	94.15±0.15	93.77±0.27	92.67±0.12	89.80±0.38
PALS	94.77±0.12	94.53±0.08	93.71±0.06	92.08±0.55

CIFAR-100	$q = 0.03$		$q = 0.05$	
	$\eta = 0.1$	$\eta = 0.3$	$\eta = 0.3$	$\eta = 0.5$
PiCO	67.38±0.09	58.64±0.28	58.18±0.60	35.51±1.14
PiCO+	70.89±0.13	64.22±0.23	62.24±0.38	45.29±0.14
ALIM-Scale	74.33±0.11	71.69±0.39	71.68±0.14	64.74±0.16
ALIM-Onehot	71.43±0.21	70.14±0.25	70.05±0.43	63.39±0.82
PALS	76.68±0.08	75.00±0.16	73.56±0.26	70.85±0.49

Table 6. Comparison of PALS with the previous state-of-the-art methods when excluding Mix-up and Consistency regularization from all the methods.

label by taking into account the annotations provided by human annotators. Table 5a presents results under the assumption of ten available human annotators, simulating the PLL scenario with a negligible noise rate, while Table 5b displays results assuming three human annotators, simulating the NPLL scenario characterized by a substantial noise rate. We compare against two noisy label learning methods: Divide-Mix [12] and ELR+ [14], a PLL method (PiCO) and an NPLL method (ALIM). In all cases, we use a pre-trained ResNet50 backbone to allow a fair one-to-one comparison. PALS achieves competitive performance on Treeversity. On Benthic and Plankton, which contain underwater images, there is a sharp decline in the performance of PiCO and ALIM. PALS displays promising generalization capability by outperforming them by a significant margin.

Mix-up	CR	$q = 0.05$		K	$q = 0.05$	
		$\eta = 0.3$	$\eta = 0.5$		$\eta = 0.3$	$\eta = 0.5$
✗	✗	73.30	71.41	10	77.87	74.51
✗	✓	74.33	72.51	15	77.61	75.15
✓	✗	75.91	73.12	25	78.29	74.98
✓	✓	77.61	75.15	50	77.01	75.60

(a) Influence of regularization

(b) Impact of K

r	0.6	0.5	0.4	0.0	δ	0.0	0.25	0.50	0.75
$q=0.05$	76.87	77.61	78.19	75.64	$q=0.05$	75.92	77.61	78.22	77.62
$\eta=0.3$					$\eta=0.3$				
$q=0.05$	74.17	75.15	74.41	65.63	$q=0.05$	73.50	75.15	74.72	72.24
$\eta=0.5$					$\eta=0.5$				

(c) Impact of label smoothing rate r

(d) Impact of δ

Backbone	$q = 0.05$		Pseudo label	$q = 0.05$	
	$\eta = 0.3$	$\eta = 0.5$		$\eta = 0.3$	$\eta = 0.5$
Wide-ResNet-34x10	81.3	78.18	✗	76.71	72.86
ResNext-50	79.55	76.37	✓	77.61	75.15
SE-ResNet-50	79.46	75.56			
DenseNet-121	78.75	74.36			
ResNet-18	77.61	75.15			

(e) Influence of backbones

(f) Impact of Eq. (1)

Table 7. Ablation studies

4.3. Ablation studies

In Table 7, we perform ablations using two different noise rates: ($\eta = 0.3$ and $\eta = 0.5$), while setting $q = 0.05$. All ablations are performed using the CIFAR-100 dataset.

Influence of regularization components: In table 7a, we study the impact of using Mix-up and CR. Removing Mix-up causes a decrease of around 2 – 3pp, and the exclusion of CR causes a decrease of about 1pp, which validates their role in our framework. We observe that their combined usage generates a more substantial effect. It must be noted that prior methods also utilize Mix-up, CR or both. For a fair comparison, we remove both Mix-up and CR from all the approaches and present the results in table 6. PALS achieves SOTA results in this setup as well and in the high noise setting of CIFAR-100 ($\eta = 0.5$), outperforms the second best method (ALIM-Scale) by a significant margin of over 6pp.

Impact of K in KNN: We vary the value of K from the set $\{10, 15, 25, 50\}$. We find that performance remains similar across different K values, with statistically insignificant differences. We use $K = 15$ for all our experiments.

Impact of δ parameter: δ controls the number of samples chosen per class to construct the reliable set. In ta-

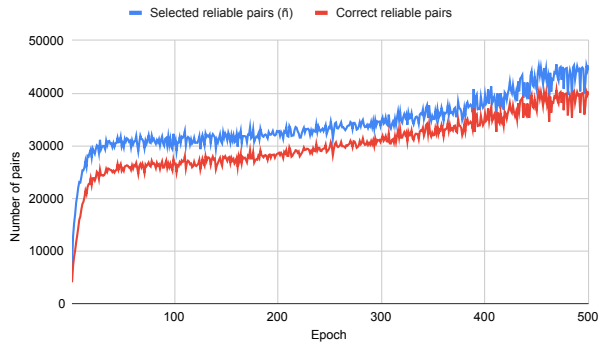


Figure 2. Number of selected reliable pairs (\tilde{n}) while training on CIFAR-100 with $q = 0.05$ and $\eta = 0.3$

ble 7d, $\delta = 0.5$ yields the best results for $\eta = 0.3$, while $\delta = 0.25$ produces the best results for $\eta = 0.5$. Lower δ implies that a smaller but more confident reliable set is constructed, which leads to better performance in a high noise setting.

Impact of label smoothing: In Table 7c, $r = 0.4$ yields the best results for $\eta = 0.3$, while $r = 0.5$ produces the best results for $\eta = 0.5$. Considering that elevated noise levels lead to increased noise in pseudo-labelling, a higher label smoothing rate appears to confer advantages. Moreover, when $r = 0$, there is a significant drop in the accuracy under the extreme noise setting. The result strongly establishes the noise-robustness imparted by label smoothing in the NPLL context.

Influence of backbones: Table 7e establishes that PALS is robust to the choice of backbone. Prior effort [29] has shown that this is not necessarily the case with other methods like PiCO.

Impact of pseudo-label computation in Eq. (1): The first row of Tab. 7f shows the results when Eq. (1) is skipped, and class posterior probabilities are computed directly based on the partial label Y_k . There is a small drop in performance, showing that initial pseudo-label computation is important for robustness against noise.

Insights into pseudo-labelling: Figure 2 shows the number of selected reliable samples \tilde{n} (blue curve), while training on CIFAR-100 with $q = 0.05$ and $\eta = 0.3$. For visualization purposes, we also plot the number of the selected examples that are correct (red curve) by comparing them with the ground truth. In the first epoch, we observe that out of the 50000 samples, 7211 samples are selected for training (of which 4036 are correct). This highlights the effectiveness of the pseudo-labelling approach, given that the backbone is randomly initialized and there is a presence of significant noise in the partial labels. Since a good portion of selected samples are correct, this aids in learning improved features. Over time, both the number of selected

examples and the proportion of correct labels increase, enhancing the quality of the feature representations. At convergence, (\tilde{n}) is 45285, of which 40330 pseudo-labels are correct.

Training time: PALS is about 1.5 times faster to train compared to PiCO+ and ALIM. On an NVIDIA GeForce RTX 2080 Ti GPU, one epoch of PALS takes 40 seconds, while one epoch of ALIM takes 60 seconds. The numbers reported in our paper are by training for 500 epochs, while ALIM reports numbers after training for 800-1000 epochs (based on their official code). So effectively, PALS is about three times faster to train. On larger models, the gap widens. For example, for a WideResNet, PALS takes 221 seconds, and ALIM takes 550 seconds per epoch, making PALS 2 times faster.

5. Conclusion

We propose PALS, a framework for NPLL that iteratively performs pseudo-labelling and classifier training. Pseudo-labelling involves the weighted KNN algorithm, and a classifier is trained using the labelled samples while leveraging label smoothing and optional regularization components. PALS achieves SOTA results in a comprehensive set of experiments, varying partial rates and noise rates. The gains are more pronounced with an increase in noise and the number of classes. Notably, PALS excels in the simulated fine-grained classification tasks. Unlike many previous methods, PALS preserves its performance as the ambiguity in the dataset increases. PALS also generalizes well to the real world, fine-grained, crowd-sourced datasets.

We provide thorough ablation studies to examine the functioning of the PALS framework. Our analysis underscores the consistent contributions of both consistency regularization and Mix-up to overall performance improvements. Additionally, we highlight the enormous benefits of label smoothing, particularly in high-noise scenarios. We believe that the PALS framework and the insights presented in this study would be helpful for the research community across a wide range of practical applications.

References

- [1] Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. *Advances in neural information processing systems*, 27, 2014. 4
- [2] Forrest Briggs, Xiaoli Z Fern, and Raviv Raich. Rank-loss support instance machines for miml instance annotation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 534–542, 2012. 1
- [3] Timothee Cour, Benjamin Sapp, Chris Jordan, and Ben Taskar. Learning from ambiguously labeled images. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 919–926. IEEE, 2009. 2

- [4] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 113–123, 2019. 5
- [5] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 5
- [6] Lei Feng, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama. Provably consistent partial-label learning. *Advances in neural information processing systems*, 33:10948–10960, 2020. 2, 5
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 5
- [8] Eyke Hüllermeier and Jürgen Beringer. Learning from ambiguously labeled examples. In *International Symposium on Intelligent Data Analysis*, pages 168–179. Springer, 2005. 1, 2
- [9] Rong Jin and Zoubin Ghahramani. Learning with multiple labels. In *Advances in Neural Information Processing Systems*. MIT Press, 2002. 2
- [10] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 5
- [11] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5
- [12] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2019. 1, 7
- [13] Zheng Lian, Mingyu Xu, Lan Chen, Licai Sun, Bin Liu, and Jianhua Tao. Irnet: Iterative refinement network for noisy partial label learning, 2023. 1, 2, 5
- [14] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342, 2020. 7
- [15] Michal Lukasik, Srinadh Bhojanapalli, Aditya Menon, and Sanjiv Kumar. Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pages 6448–6458. PMLR, 2020. 4
- [16] Jiaqi Lv, Miao Xu, Lei Feng, Gang Niu, Xin Geng, and Masashi Sugiyama. Progressive identification of true labels for partial-label learning. In *International conference on machine learning*, pages 6500–6510. PMLR, 2020. 2
- [17] Jiaqi Lv, Biao Liu, Lei Feng, Ning Xu, Miao Xu, Bo An, Gang Niu, Xin Geng, and Masashi Sugiyama. On the robustness of average losses for partial-label learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3
- [18] Diego Ortego, Eric Arazo, Paul Albert, Noel E O’Connor, and Kevin McGuinness. Multi-objective interpolation training for robustness to label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6606–6615, 2021. 3
- [19] Congyu Qiao, Ning Xu, Jiaqi Lv, Yi Ren, and Xin Geng. Fre-dis: A fusion framework of refinement and disambiguation for unreliable partial label learning. In *International Conference on Machine Learning*, pages 28321–28336. PMLR, 2023. 1, 2, 5
- [20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015. 5
- [21] Lars Schmarje, Vasco Grossmann, Claudius Zelenka, Sabine Dippel, Rainer Kiko, Mariusz Oszust, Matti Pastell, Jenny Stracke, Anna Valros, Nina Volkmann, et al. Is one annotation enough?-a data-centric image classification benchmark for noisy and ambiguous label estimation. *Advances in Neural Information Processing Systems*, 35:33215–33232, 2022. 2, 5, 6
- [22] Yu Shi, Ning Xu, Hua Yuan, and Xin Geng. Unreliable partial label learning with recursive separation. In *IJCAI*, pages 4208–4216, 2023. 1, 2, 5
- [23] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020. 1
- [24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 2, 4
- [25] Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. PiCO: Contrastive label disambiguation for partial label learning. In *International Conference on Learning Representations*, 2022. 2, 5
- [26] Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. Pico+: Contrastive label disambiguation for robust partial label learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 1, 2, 3, 5
- [27] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010. 5
- [28] Hongwei Wen, Jingyi Cui, Hanyuan Hang, Jiabin Liu, Yisen Wang, and Zhouchen Lin. Leveraged weighted loss for partial label learning. In *International Conference on Machine Learning*, pages 11091–11100. PMLR, 2021. 2, 5
- [29] Dong-Dong Wu, Deng-Bao Wang, and Min-Ling Zhang. Revisiting consistency regularization for deep partial label learning. In *International Conference on Machine Learning*, pages 24212–24225. PMLR, 2022. 2, 5, 8
- [30] Shiyu Xia, Jiaqi Lv, Ning Xu, Gang Niu, and Xin Geng. Towards effective visual representations for partial-label learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15589–15598, 2023. 2
- [31] Mingyu Xu, Zheng Lian, Lei Feng, Bin Liu, and Jianhua Tao. Alim: Adjusting label importance mechanism for noisy

- partial label learning. *Advances in Neural Information Processing Systems*, 36, 2024. [1](#), [3](#), [5](#)
- [32] Yao Yao, Jiehui Deng, Xiuhua Chen, Chen Gong, Jianxin Wu, and Jian Yang. Deep discriminative cnn with temporal ensembling for ambiguously-labeled image classification. In *Proceedings of the aaai conference on artificial intelligence*, pages 12669–12676, 2020. [2](#)
- [33] Fei Yu and Min-Ling Zhang. Maximum margin partial label learning. In *Asian conference on machine learning*, pages 96–111. PMLR, 2016. [2](#)
- [34] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. [4](#)
- [35] Min-Ling Zhang and Fei Yu. Solving the partial label learning problem: An instance-based approach. In *IJCAI*, pages 4048–4054, 2015. [2](#)
- [36] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering*, 26(8):1819–1837, 2013. [1](#)