

Mixed Blessing: Class-Wise Embedding guided Instance-Dependent Partial Label Learning

Fuchao Yang

yangfc@seu.edu.cn

College of Software Engineering,
Southeast University
Nanjing, China

Yongqiang Dong

dongyq@seu.edu.cn

School of Computer Science and
Engineering, Southeast University
Nanjing, China

Jianhong Cheng

chengjh@seu.edu.cn

School of Computer Science and
Engineering, Southeast University
Nanjing, China

Hui Liu

h2liu@sfu.edu.hk

Yam Pak Charitable Foundation
School of Computing and Information
Sciences, Saint Francis University
Hong Kong, China

Junhui Hou

jh.hou@cityu.edu.hk

Department of Computer Science,
City University of Hong Kong
Hong Kong, China

ABSTRACT

In partial label learning (PLL), every sample is associated with a candidate label set comprising the ground-truth label and several noisy labels. The conventional PLL assumes the noisy labels are randomly generated (instance-independent), while in practical scenarios, the noisy labels are always instance-dependent and are highly related to the sample features, leading to the instance-dependent partial label learning (IDPLL) problem. Instance-dependent noisy label is a double-edged sword. On one side, it may promote model training as the noisy labels can depict the sample to some extent. On the other side, it brings high label ambiguity as the noisy labels are quite undistinguishable from the ground-truth label. To leverage the nuances of IDPLL effectively, for the first time we create class-wise embeddings for each sample, which allow us to explore the relationship of instance-dependent noisy labels, i.e., the class-wise embeddings in the candidate label set should have high similarity, while the class-wise embeddings between the candidate label set and the non-candidate label set should have high dissimilarity. Moreover, to reduce the high label ambiguity, we introduce the concept of class prototypes containing global feature information to disambiguate the candidate label set. Extensive experimental comparisons with twelve methods on six benchmark data sets, including four fine-grained data sets, demonstrate the effectiveness of the proposed method. The code implementation is publicly available at <https://github.com/Yangfc-ML/CEL>.

CCS CONCEPTS

- Computing methodologies → Learning paradigms.

*Corresponding Author

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, July 2017, Washington, DC, USA

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
<https://doi.org/XXXXXXX.XXXXXXX>

Yuheng Jia*

yhjia@seu.edu.cn

School of Computer Science and
Engineering, Southeast University
Nanjing, China

KEYWORDS

weakly supervised learning, partial label learning, instance-dependent partial label learning

ACM Reference Format:

Fuchao Yang, Jianhong Cheng, Hui Liu, Yongqiang Dong, Yuheng Jia, and Junhui Hou. 2018. Mixed Blessing: Class-Wise Embedding guided Instance-Dependent Partial Label Learning. In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 10 pages. <https://doi.org/XXXXXXX.XXXXXXX>

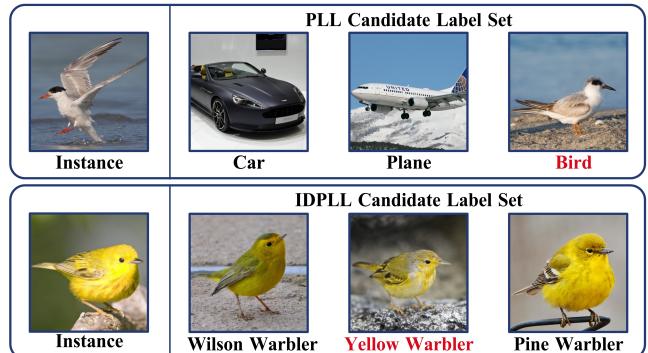


Figure 1: Differences between the conventional PLL and IDPLL, where the red label is the ground-truth label of the instance. In PLL, the noisy labels in the candidate label set are randomly generated. However, in IDPLL, the noisy labels in the candidate label set are instance-dependent, making them very similar to that of the ground-truth label, which brings more label ambiguity.

1 INTRODUCTION

Partial label learning (PLL) [10, 12, 21, 27] has garnered significant attention over the past decade as a form of weakly supervised learning. In PLL, each sample is associated with a candidate label set, concealing the ground-truth label and several noisy labels

within it. In the training phase, the ground-truth label is inaccessible. PLL does not rely on precise labeling, leading to substantial reductions in time and resource costs associated with sample annotation. Consequently, PLL finds extensive applications across various real-world domains, including automatic image annotation [1], web mining [20], ecoinformatics [18], and multimedia content analysis [39]. The crux of addressing the PLL problem lies in label disambiguation, involving the identification of the ground-truth label while mitigating the impact of noisy labels within the candidate label set. Conventional label disambiguation techniques [22, 30, 33] have demonstrated strong efficacy in the typical PLL scenarios where noisy labels are randomly generated (instance-independent) [11, 29].

Recently, a more realistic PLL framework called instance-dependent partial label learning (IDPLL) [8, 34, 38] was proposed, where noisy labels are re-tailored to individual samples (instance-dependent). As shown in Fig. 1, in the conventional PLL, the noisy labels in the candidate label set may have significant differences with the ground-truth label because the noisy labels are randomly generated. However, in IDPLL, the noisy labels are very similar to the ground-truth label. To be more specific, these birds all have yellow feathers, which is more easily to cause label ambiguity. Conventional PLL methods often struggle in IDPLL since they ignore the characteristics of IDPLL.

IDPLL is a mixed blessing. On the positive side, as shown in Fig. 2, the model achieves better classification performance in the early stage and converges faster in the IDPLL setting compared with the PLL setting. This is because in IDPLL the noisy labels in the candidate label set are related to the sample, which can describe the sample to some extent. Although the noisy labels in the candidate label set are incorrect, they can still act as supervision to promote model training. On the negative side, the model only achieves quite inferior classification performance in the later stage. The reason is that given the instance, the instance-dependent noisy labels in the candidate label set are highly similar to the ground-truth label, bringing more label ambiguity and making the candidate label disambiguation process more challenging. The current IDPLL methods have made different attempts to achieve better label disambiguation. For example, NEPLL [8] proposed a well-disambiguated sample selection method based on normalized cross-entropy and trained the model progressively according to the selected samples. POP [37] proposed to purify the candidate label set during the training phase to gradually reduce the difficulty of label disambiguation. DIRK [34] proposed a label rectification strategy that ensured the model output on the candidate label set was always higher than that on the non-candidate label set. However, these methods did not comprehensively exploit both the positive side and negative side of IDPLL, limiting their performance.

To address this double-edged sword challenge in IDPLL, we propose a novel method called **CEL** (Mixed Blessing: Class-Wise Embedding guided Instance-Dependent Partial Label Learning). Specifically, unlike previous PLL and IDPLL methods, where each sample corresponds to a single embedding, our method introduces the class-wise embedding for each sample, i.e., each sample has multiple embeddings corresponding to different classes, to comprehensively exploit the mixed blessing in IDPLL. First, we propose a class associative loss (CAL) to leverage the relationships among

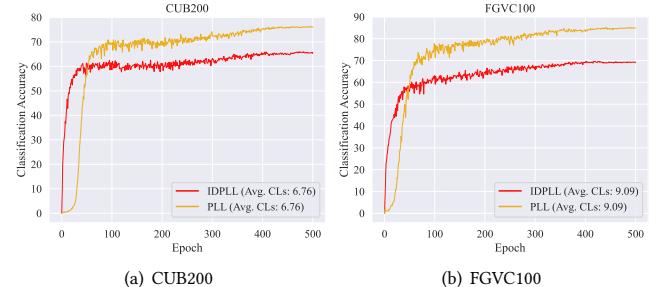


Figure 2: The classification accuracy curves of PLL method PRODEN [22] in IDPLL and PLL settings on two data sets CUB200 and FGVC100, where AVG. CLs represent the average number of candidate labels of each sample. In the IDPLL setting, the model has a faster learning speed in the early stage of training because the instance-dependent noisy labels are related to the sample to some extent, which can provide more supervision in the early stage of model training. However, in the later stage of training, the performance of PRODEN in the IDPLL setting is significantly inferior to that in the PLL setting as instance-dependent noisy labels bring more label ambiguity.

different classes of each sample to guide the learning of class-wise embeddings. In IDPLL, the class-wise embeddings within the candidate label set should exhibit high similarity to each other as the candidate labels can describe the instance to some extent. In contrast, the class-wise embeddings between the candidate label set and the non-candidate label set should display stark differences. In this way, we can obtain class-wise embeddings that are more suitable for IDPLL. Second, we propose a prototype discriminative loss (PDL) by constructing class prototype for each class which contains global feature information to guide the label disambiguation process. To be specific, we select the most high-confidence label for each sample based on the model output, and then we ensure the class-wise embedding of this particular high-confidence label is aligned with its corresponding class prototype while being distanced from other class prototypes, thereby enhancing the model's discriminative ability. By employing the above two losses, we can obtain embeddings that are tailor-made for IDPLL and enhance the model's label disambiguation performance simultaneously, thus addressing the mixed blessing issue in IDPLL. Extensive experiments on 6 benchmark data sets demonstrate the effectiveness of the proposed method when compared with 12 state-of-the-art methods.

The contributions of our work are summarized as follows:

- To the best of our knowledge, we are the first to introduce class-wise embedding in IDPLL. Class-wise embedding enable the model to explore the nuances relationships of classes in each sample, thereby better addressing the mixed blessing problem inherent in IDPLL.
- We comprehensively consider the positive and negative sides of IDPLL. To leverage the positive side, we utilize the class associative loss to exploit the relationships among the labels

- (including both candidate labels and non-candidate labels) of each sample. To address the negative side, we apply the prototype discriminative loss to utilize the relationships between high-confidence class and class prototypes.
- Extensive experiments on benchmark data sets demonstrate that our method achieves significantly superior performance when comparing with both PLL and IDPLL methods.

2 RELATED WORK

2.1 Partial Label Learning

Conventional PLL methods can be roughly divided into two categories based on the used information. The first category uses the information in the feature space to guide label disambiguation [3, 9, 29, 40, 41]. The second category leverages the information of label space to achieve label disambiguation [4, 11, 17]. These methods often rely on linear models and are difficult to apply to large-scale data sets. Deep PLL has received wide attention in recent years and it adopts deep neural networks to process large-scale data sets [22, 33, 36]. PRODEN [22] proposed to progressively identify the ground-truth label during the self-training procedure. RC and CC [5] proposed provably risk-consistent and classifier-consistent label disambiguation methods. LWS [32] proposed a family of leveraged weighted loss functions. PICO [30] introduced the widely used contrastive loss [6] to the PLL, which became the foundation for a large number of follow-up works. CR-DPLL [33] employed consistency regularization to reduce the impact of noisy labels. PAPI [36] constructed the similarity score between feature prototypes and sample embeddings, and improves the model performances by aligning the similarity score with the model output. CROSEL [26] used two models to cross-select trustworthy samples from the data set for the training phase. The above methods have achieved superior results in the conventional PLL setting, where the noisy labels in the candidate label set are randomly generated. However, in practice, noisy labels are always instance-dependent [24, 35, 38]. For example, in crowdsourced annotation tasks, the annotations from many annotators constitute the candidate label set of each sample, and the noisy labels within it are all instance-dependent. Consequently, the performances of conventional PLL methods are often limited due to the lack of consideration of the characteristics of IDPLL.

2.2 Instance-Dependent Partial Label Learning

IDPLL is a PLL framework that is closer to real-world scenarios. VALEN [38] was the first work to introduce the concept of IDPLL, with a two-stage disambiguation process. The first stage involves recovering samples' latent label distribution, and then training the model using the recovered label distribution in the second stage. ABLE [35] proposed an ambiguity-induced positive selection contrastive learning framework for IDPLL to achieve label disambiguation. NEPLL [8] introduced a sample selection method based on normalized entropy, intending to separate well-disambiguated samples and under-disambiguated samples. It also established a dynamic candidate-aware thresholding for the further refinement of sample selection. POP [37] presented a method that progressively purified the candidate label set and optimized the classifier.

IDGP [24] modeled the candidate label generation process for IDPLL, simulating the ground-truth label and noisy labels generation processes with Categorical distribution and Bernoulli distribution respectively. DIRK [34] proposed a self-distillation-based label disambiguation method, i.e., the training of the student model is directed by the output of the teacher model. It also developed a label confidence rectification method that meets the prior knowledge of IDPLL. However, the aforementioned IDPLL methods did not consider that the IDPLL setting is a mixed blessing, i.e., on the positive side, noisy labels can describe the sample to some extent and help the model training, while on the negative side, identifying the ground-truth label within the candidate label set becomes more challenging. They only focused on developing better disambiguation methods while ignoring the noisy label information, thereby limiting their performance.

3 PROPOSED METHOD

3.1 Class-Wise Embedding

Denote by $\mathcal{X} \subset \mathbb{R}^d$ the d -dimensional feature space and $\mathcal{Y} = \{1, 2, \dots, q\}$ the corresponding label space with q classes. Let $\mathcal{D} = \{(x_i, S_i), 1 \leq i \leq m\}$ denote a partial label data set comprising m samples, where S_i is the candidate label set of sample x_i and the ground-truth label of sample i is concealed in S_i . The objective of IDPLL is to induce a multi-class classifier that maps elements from \mathcal{X} to \mathcal{Y} . Previous models in the realm of PLL and IDPLL typically consist of two modules. The first module is the backbone responsible for deriving the feature map $M \in \mathbb{R}^{H \times W \times C}$ for each sample, where H , W and C are the height, width, and channel respectively. Afterward, M is processed through average global pooling to obtain the low dimensional embedding. The second module encompasses a linear layer that translates this low-dimensional embedding into the final prediction.

In this paper, we use a different paradigm that consists of three modules. As shown in Fig. 3, the first module is as same as other models, i.e., a backbone f , which extracts the feature map $M_i = f(x_i) \in \mathbb{R}^{H \times W \times C}$ of each sample x_i through the deep neural network. The second module is a **class-wise embedding** encoder [16, 19, 25] g which produces class-wise embeddings $E_i = g(M_i) \in \mathbb{R}^{q \times l}$, where l is the length of each class-wise embedding. Note, the E_i^j indicates the representation of the j -th class of sample x_i . The third module is a group of linear layers z that output the predicted probabilities $P_i = z(E_i) \in \mathbb{R}^q$.

In conventional PLL and IDPLL representation methods, the features of each sample are extracted as a single embedding, so they can only consider relationships at the sample level. However, in IDPLL, the relationships between each sample's candidate labels and non-candidate labels are valuable and worth leveraging. By employing class-wise encoder, we can represent each sample's embeddings on different labels. This allows us to explore the internal relationships among different classes and fully utilize the prior knowledge of IDPLL, making it a more suitable representation method for IDPLL.

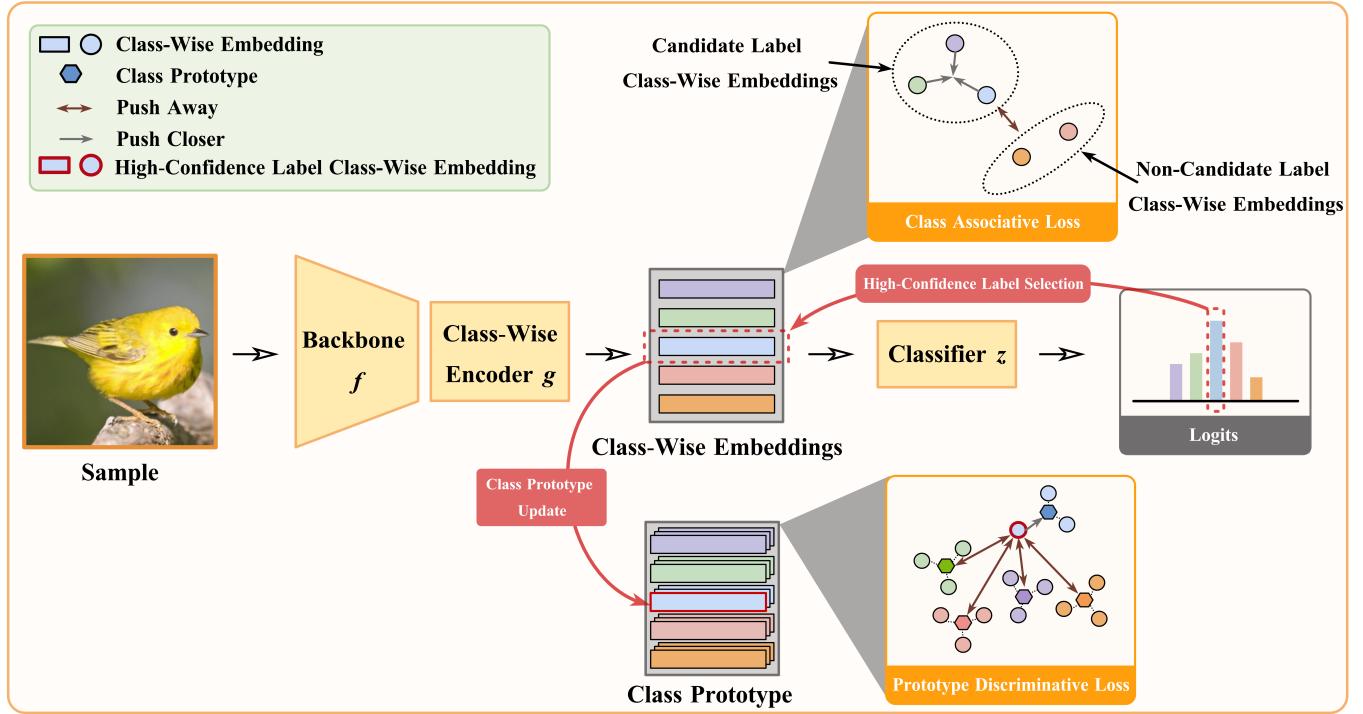


Figure 3: Illustration of our method CEL. Our model consists of three modules: the backbone f , the class-wise encoder g , and the classifier z . For each sample, after processing through the backbone f and class-wise encoder g , each sample obtains its class-wise embeddings, i.e., each class corresponds to an embedding for that sample. The red line represents the process of constructing class prototypes based on the high-confidence class selected according to the model’s output. The class associative loss (CAL) considers the relationships among each sample’s different class-wise embeddings. While the prototype discriminative loss (PDL) considers the relationships between high-confidence class and class prototypes.

3.2 Label Disambiguation Loss

As aforementioned, the pivotal process in addressing PLL is label disambiguation, which mitigates the impact of noisy labels within the candidate label set. Here, we adopt a widely used deep PLL disambiguation strategy [22], which constructs sample label confidences based on model outputs during training. Initially, the label confidence vector $T_i \in \mathbb{R}^q$ of sample x_i is initialized as $T_{ij} = \frac{1}{|S_i|}$, if $j \in S_i$ and $T_{ij} = 0$, otherwise, where $|S_i|$ returns the number of candidate labels of sample x_i . Throughout the training, we update the label confidence according to the model output:

$$T_{ij} = \begin{cases} \frac{P_i^j}{\sum_{k \in S_i} P_i^k}, & \text{if } j \in S_i, \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where P_i^j represents the model prediction on j -th class of sample x_i . The goal of Eq. (1) is to eliminate the influence of non-candidate labels, so that the model can focus on the candidate labels. Based on the disambiguated confidence, we can obtain the following classification loss to guide the model training:

$$\mathcal{L}_{cls} = \frac{1}{m} \sum_{i=1}^m \ell(P_i, T_i), \quad (2)$$

where ℓ indicates the cross-entropy loss. Eq. (2) is usually viewed as a self-learning process and has demonstrated efficacy across various PLL scenarios.

3.3 Class Associative Loss based on Class-Wise Embedding

As previous analyzed, IDPLL is a double-edged sword. On the positive side, the noisy labels in the candidate label set are very similar to the ground-truth label because they often share common features and they can depict the sample to some extent, which is also the fundamental reason leading to label ambiguity. Therefore, an important characteristic of IDPLL is that the labels within the candidate label set should be very similar, while the labels in the candidate label set should exhibit significant differences from the labels in the non-candidate label set. To fully leverage this prior knowledge, we can measure the relationships among classes of each sample through class-wise embeddings. To be specific, the class-wise embeddings corresponding to each label in the candidate label set should be similar to each other, while the class-wise embeddings between the candidate label set and the non-candidate label set should display stark differences. Therefore, we can construct the

following class-wise relationships:

$$s_i^{cal} = \sum_{j,k \in S_i} \langle E_i^j, E_i^k \rangle / \sum_{j,k \in S_i} 1, \quad (3)$$

$$d_i^{cal} = \sum_{j \in S_i, h \notin S_i} \langle E_i^j, E_i^h \rangle / \sum_{j \in S_i, h \notin S_i} 1, \quad (4)$$

where E_i^j and S_i represent the j -th class-wise embedding and the candidate label set of sample x_i respectively. $\langle \cdot, \cdot \rangle$ returns the cosine similarity of two vectors. Note that the class-wise embeddings have been normalized before calculating the cosine similarity. s_i^{cal} in Eq. (3) denotes the average similarity of classes in the candidate label set, while d_i^{cal} in Eq. (4) indicates the average similarity of classes between the candidate label set and the non-candidate label set.

By considering the similarities of Eqs. (3) and (4) simultaneously, we can obtain the following class associative loss (CAL) function:

$$\mathcal{L}_{cal} = \frac{1}{m} \sum_{i=1}^m \left(\left(1 - s_i^{cal} \right) + \gamma_1 |d_i^{cal}| \right), \quad (5)$$

where γ_1 is a trade-off parameter that balances two different terms. $|\cdot|$ is the absolute value operator given $d_i^{cal} \in (-1, 1)$. The first term in Eq. (5) means that for each sample x_i , the class-wise embeddings in its candidate label set should be pulled as close as possible. In the meanwhile, the second term implies that the class-wise embeddings between the candidate label set and the non-candidate label set should be pushed as far away as possible, and in the ideal situation, their class-wise embeddings should be orthogonal, i.e. $d_i^{cal} = 0$. By minimizing the loss function \mathcal{L}_{cal} , we can obtain a model that is more suitable for the IDPLL setting.

3.4 Prototype Discriminative Loss based on Class-Wise Embedding

In IDPLL, we can distinguish the labels between the candidate label set and the non-candidate label set easily, however, it becomes more difficult to identify which label in the candidate label set is the ground-truth label, as the candidate labels are similar to each other, bringing more label ambiguity. Therefore, to tackle this negative side of IDPLL and identify the ground-truth label, we use the global information of samples to guide the model's disambiguation. Therefore, we first construct prototypes for each class,

$$Q^c = \text{Normalize}(Q^c + E_i^c), \text{ if } c = \text{argmax}(P_i) \text{ and } c \in S_i, \quad (6)$$

where Q^c is the c -th class prototypes and $\text{Normalize}(\cdot)$ denotes the L_2 normalization operator. To ensure the quality of the class prototypes, we only select the class with the highest model output probability in the candidate label set as the high-confidence label of each sample and add the corresponding class-wise embedding into the class prototype.

Similar to Eq. (3) and Eq. (4), we construct the similarity relationships between class-wise embeddings and class prototypes as follows:

$$s_i^{pd़l} = \langle E_i^c, Q^c \rangle, \text{ if } c = \text{argmax}(P_i) \text{ and } c \in S_i, \quad (7)$$

$$d_i^{pd़l} = \sum_{k \neq c, k \in [q]} \langle E_i^c, Q^k \rangle / (q - 1), \quad (8)$$

Algorithm 1: The Pseudo Code of the Proposed Method

Input:

\mathcal{D} : the instance-dependent partial label training set;

T_{max} : the total training epoch;

T_w : the first stage epoch;

$\alpha, \beta, \gamma_1, \gamma_2$: the parameters of the loss function;

Output:

model parameters.

```

1 for  $t = 1$  to  $T_{max}$  do
2   Sample a mini-batch from  $\mathcal{D}$ ;
3   if  $t \leq T_w$  then
4     | Calculate the loss according to Eq. (10);
5   else
6     | Calculate the loss according to Eq. (11);
7   end
8   Update the label confidence according to Eq. (1);
9   Update the class prototype according to Eq. (6);
10  Update the model parameters;
11 end
12 Return result.

```

where $s_i^{pd़l}$ in Eq. (7) represents the similarity between the class-wise embedding of the class with the highest model prediction in the candidate label set and the corresponding class prototype. Meanwhile, $d_i^{pd़l}$ in Eq. (8) denotes the average similarity between this selected class-wise embedding and all other class prototypes. By combining Eq. (7) and Eq. (8), we have the following prototype discriminative loss (PDL) function:

$$\mathcal{L}_{pd़l} = \frac{1}{m} \sum_{i=1}^m \left(\left(1 - s_i^{pd़l} \right) + \gamma_2 |d_i^{pd़l}| \right), \quad (9)$$

where γ_2 is the trade-off parameter that balances the two different prototype level terms. By minimizing Eq. (9), we can guide the model training through class prototypes, i.e., the most reliable class predicted by the model should be as close as possible to the corresponding class prototype, while this reliable class should be far away from other class prototypes, and in the most ideal case, their similarity relationship should be 0. By utilizing the global information of class prototypes, we can effectively improve the discriminative performance of the model and select the ground-truth label from the candidate label set.

3.5 Overall Objective

Considering the low quality of class prototypes obtained in the early stages of model training, it is difficult to ensure the effectiveness of label disambiguation. Therefore, we divide the model training into two stages. In the first stage, our training uses the classification loss and class associative loss which aims to learn a model more suitable for IDPLL. The training objective of the first stage can be written as follows:

$$\mathcal{L}_{all} = \mathcal{L}_{cls} + \alpha \mathcal{L}_{cal}. \quad (10)$$

After training for T_w epochs, we add the prototype discriminative loss into the training objective to further improve the model's

Table 1: Characteristic of the benchmark data sets, where Avg. CLs represent the average number of candidate labels per sample.

	Data set	Train	Test	Dimensions	Classes	Avg. CLs
Common	CIFAR-100	50000	10000	32×32	100	10.82 (rate=0.1)
	CIFAR-100H	50000	10000	32×32	100	3.41 (rate=0.6)
Fine-Grained	FGVC Aircraft (FGVC100)	6776	3333	224×224	100	9.09 (rate=0.1)
	Stanford Dogs (DOGS120)	12000	8580	224×224	120	11.51 (rate=0.1)
	Stanford Cars (CARS196)	8144	8041	224×224	196	9.98 (rate=0.05)
	CUB-200-2011 (CUB200)	5994	5794	224×224	200	6.76 (rate=0.03)

Table 2: Classification accuracy (mean±std) of each method on benchmark data sets. Bold and underlined indicate the best and second-best results, respectively.

Type	Method	CIFAR-100	CIFAR-100H	FGVC100	DOGS120	CARS196	CUB200
	CEL (OURS)	75.51±0.28%	75.77±0.09%	78.36±0.19%	78.18±0.12%	86.22±0.08%	68.60±0.10%
	DIRK (AAAI 2024)	<u>74.40±0.18%</u>	<u>75.50±0.45%</u>	76.86±3.54%	<u>75.97±0.29%</u>	85.31±0.77%	<u>66.60±1.07%</u>
	NEPLL (ICCV 2023)	72.41±0.66%	75.05±0.83%	75.36±0.59%	74.84±0.09%	85.05±0.17%	62.88±1.66%
IDPLL	POP (ICML 2023)	72.74±0.70%	75.11±0.18%	<u>77.87±0.23%</u>	74.86±0.12%	85.26±0.24%	64.88±0.48%
	IDGP (ICLR 2023)	68.49±0.35%	68.83±1.11%	72.48±0.86%	66.79±0.38%	79.56±0.46%	58.16±0.58%
	ABLE (IJCAI 2022)	70.94±0.17%	73.15±0.15%	74.05±0.43%	72.78±0.07%	<u>85.75±0.24%</u>	63.23±0.36%
	VALEN (NeurIPS 2021)	68.33±0.36%	70.52±0.24%	68.21±0.43%	66.89±0.15%	82.21±0.16%	63.05±3.32%
PLL	PICO (ICLR 2022)	64.00±0.29%	65.39±0.38%	63.52±0.94%	67.80±0.06%	70.15±1.63%	58.56±0.90%
	CR-DPLL (ICML 2022)	71.54±0.25%	75.49±0.30%	63.20±1.22%	61.41±0.82%	68.30±0.45%	50.26±0.34%
	LWS (ICML 2021)	71.64±0.32%	74.40±0.62%	72.82±0.44%	66.12±0.12%	82.11±0.24%	54.01±0.68%
	PRODEN (ICML 2020)	70.49±0.50%	72.90±0.47%	69.34±0.39%	70.94±0.43%	83.35±0.05%	65.06±0.14%
	RC (NeurIPS 2020)	69.96±0.01%	72.69±0.30%	72.72±0.22%	70.40±0.26%	82.11±0.63%	60.96±0.59%
	CC (NeurIPS 2020)	70.51±0.28%	73.83±0.19%	64.36±0.50%	66.21±0.77%	70.61±2.01%	56.79±0.71%

disambiguation performance. The objective function of the second stage can be summarized as follows:

$$\mathcal{L}_{all} = \mathcal{L}_{cls} + \alpha \mathcal{L}_{cal} + \beta \mathcal{L}_{pdl}, \quad (11)$$

where α and β are two trade-off parameters. The overall pseudo-code of our method is summarized in **Algorithm 1**.

4 EXPERIMENTS

4.1 Experimental Setting

4.1.1 Data sets. To demonstrate the effectiveness of our method, we conducted comparisons on several challenging data sets with at least 100 classes. To be specific, we conducted experiments on two common image data sets including CIFAR-100 [15], CIFAR-100H [15] and four fine-grained [31, 42] image data sets including CUB-200-2011 [28], Stanford Cars [14], FGVC Aircraft [23], Stanford Dogs [13] which are more easily to cause label ambiguity. Table 1 records the detailed information of all the data sets, where Avg. CLs represent the average number of candidate labels per sample. For data sets CIFAR-100 and CIFAR-100H, the image size was set to

32 × 32, while for fine-grained data sets, we resized the images to 224 × 224. We employed the IDPLL noisy label generation method proposed by VALEN [38] to generate instance-dependent noisy labels. Note that for CIFAR-100H, noisy labels only appear in other subclasses that belong to the same superclass as the ground-truth label.

4.1.2 Comparing methods. To demonstrate the effectiveness of the proposed method, we compared our method with 12 methods including 6 IDPLL methods and 6 PLL methods. *IDPLL methods*: (i) DIRK [34], a self-distillation based label disambiguation method. (ii) NEPLL [8], a normalized entropy guided sample selection method. (iii) POP [37], a method that progressive purifies candidate label set and refines classifier. (iv) IDGP [24], a generation method that models the candidate label generation process. (v) ABLE [35], a contrastive learning-based framework that uses ambiguity-induced positives selection method. (vi) VALEN [38], a label enhancement guided latent label distribution recovery method. *PLL methods*: (i) PICO [30], a method that combines PLL and contrastive learning for the first time. (ii) CR-DPLL [33], a consistency regularization label

Table 3: Win/tie/loss counts on the classification performance of our method CEL against each comparing method on benchmark data sets according to the pairwise t-test at 0.05 significance level.

	DIRK	NEPLL	POP	IDGP	ABLE	VALEN	PICO	CR-DPLL	LWS	PRODEN	RC	CC	Total
Common	1/1/0	1/1/0	1/1/0	2/0/0	2/0/0	2/0/0	2/0/0	1/1/0	2/0/0	2/0/0	2/0/0	2/0/0	20/4/0
Fine-Grained	2/2/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	4/0/0	46/2/0
Total	3/3/0	5/1/0	5/1/0	6/0/0	6/0/0	6/0/0	6/0/0	5/1/0	6/0/0	6/0/0	6/0/0	6/0/0	66/6/0

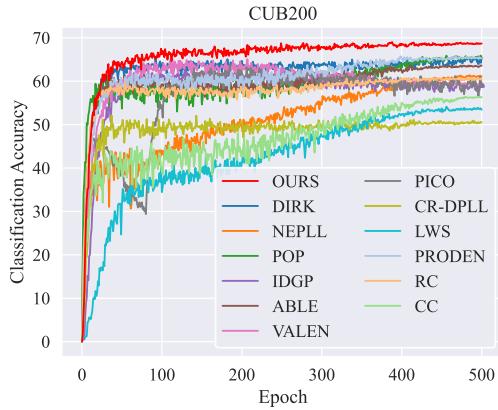


Figure 4: Classification accuracy curves of all methods on benchmark data set CUB200.

disambiguation method. (iii) LWS [32], a method that uses leveraged weighted loss to balance candidate label set and non-candidate label set. (iv) PRODEN [22], a method that progressively identifies the ground-truth label during the self-training procedure. (v) RC [5], a risk-consistent weighting method. (vi) CC [5], a classifier-consistent that uses a transition matrix. The parameters of all methods were set according to their original papers.

4.1.3 Implementation details. For fair comparisons, we employed ResNet-18 [7] as the backbone f on data sets CIFAR-100 and CIFAR-100H, while using a ResNet-34 [7] pre-trained on ImageNet [2] as the backbone f on fine-grained data sets for all methods. We used ML-Decoder [25] as our class-wise encoder g . Stochastic gradient descent (SGD) was used as the optimizer with a momentum of 0.9 and all methods were trained for 500 epochs. We selected learning rate in $\{0.5, 0.1, 0.05, 0.01, 0.005, 0.001\}$ and weight decay in $\{0.001, 0.0005, 0.0001\}$ respectively. Additionally, we applied the cosine annealing learning rate schedule for all methods. The batch sizes of all data sets were set to 128. We repeated the experiments three times under the same random seeds and recorded the mean accuracy and standard deviation. As for our method, parameters α and β were selected from $\{0.1, 0.5, 1\}$, while γ_1 and γ_2 were selected from $\{0.5, 1, 2, 5\}$. The T_w was set to 250, which is half of the total training epochs. The length of each class-wise embedding l was set to 512 on all data sets. Following DIRK [34], we used the same weak augmentation and strong augmentation for our method.

4.2 Experimental Results

4.2.1 Classification performance. Table 2 reports the classification accuracies of all methods on two common data sets and four fine-grained data sets. According to Table 2, our method ranks first in all benchmark data sets when compared with 6 PLL methods and 6 IDPLL methods. It is worth noting that there is only minor difference in classification accuracies between the previous PLL and IDPLL methods on common data sets like CIFAR-100 and CIFAR-100H. While our method improves classification accuracy from $74.40 \pm 0.18\%$ to $75.51 \pm 0.28\%$ on the CIFAR-100 data set compared to the best previous method. On fine-grained data sets, the performance gaps between PLL and IDPLL methods are significant. PLL methods generally fail to achieve satisfactory accuracy because all labels in the fine-grained data sets belong to the same superclass, which is more challenging. Consequently, IDPLL methods tailored to this setting often outperform the conventional PLL methods. Our method CEL constantly excels on these fine-grained data sets, improving classification accuracy from $66.60 \pm 1.07\%$ to $68.60 \pm 0.10\%$ on the data set CUB200 and from $75.97 \pm 0.29\%$ to $78.18 \pm 0.12\%$ on the data set DOGS120, compared to the best previous method.

4.2.2 Significance test. Table 3 reports win/tie/loss counts of our method CEL against each comparing method based on the pairwise t-test at 0.05 significance level. As shown in Table 3, on common data sets, our method CEL significantly outperforms the comparing methods in 83.3% (20/24). While on fine-grained data sets, our method significantly outperforms the comparing methods in 95.8% (46/48). Furthermore, our method achieves superior performance than all comparing methods except DIRK on fine-grained data sets. Considering all benchmark data sets, our method significantly outperformed the comparing methods in 91.7% (66/72), demonstrating the effectiveness of our method.

4.2.3 Classification accuracy curves. Fig. 4 shows the classification accuracy curves of all methods on data set CUB200. As shown in Fig. 4, compared to other methods, our method CEL maintains a relatively fast learning speed in the early stages, only slightly slower than POP, demonstrating that our class associative loss can enhance the model's learning speed. Moreover, CEL achieves the best classification accuracy in the later stages of training, proving that our prototype discriminative loss further improves the model's disambiguation performance.

4.3 Further Analysis

4.3.1 Ablation study. In Table 4, we conduct ablation studies on all benchmark data sets to demonstrate the necessity of each component in our method. The first row represents our method with only

Table 4: Ablation study of our method on the benchmark data sets, where \mathcal{L}_{cls} , \mathcal{L}_{cal} and \mathcal{L}_{pdl} indicate the label disambiguation loss, the class associative loss and the prototype discriminative loss respectively.

\mathcal{L}_{cls}	\mathcal{L}_{cal}	\mathcal{L}_{pdl}	CIFAR-100	CIFAR-100H	FGVC100	DOGS120	CARS196	CUB200	AVERAGE
✓			73.70±0.08%	75.21±0.11%	76.65±0.33%	74.83±0.75%	84.69±0.87%	65.99±0.99%	75.18%
✓	✓		74.68±0.12%	75.61±0.06%	77.71±0.17%	77.33±1.03%	85.74±0.10%	67.32±1.04%	76.40%
✓	✓	✓	75.51±0.28%	75.77±0.09%	78.36±0.19%	78.18±0.12%	86.22±0.08%	68.60±0.10%	77.11%

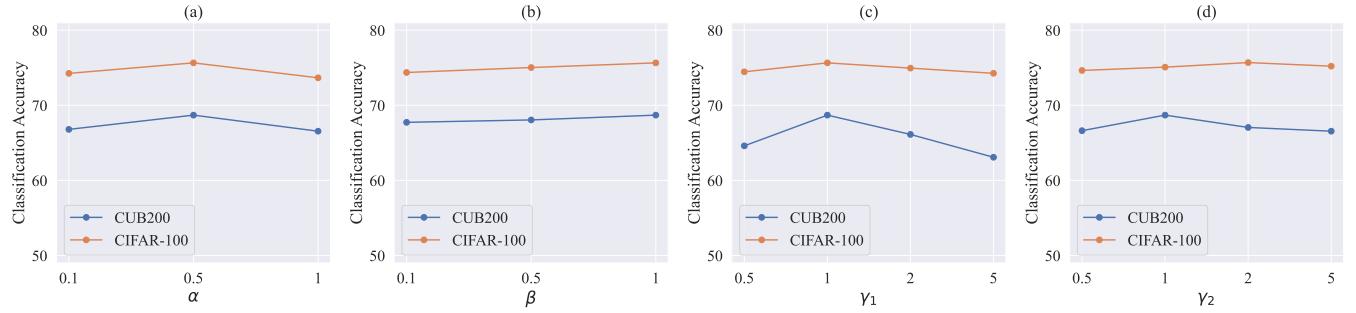


Figure 5: Parameters sensitivity of our method CEL. (a) - (d) represent the classification accuracy of our method on benchmark data sets CIFAR-100 and CUB200 by varying α , β , γ_1 and γ_2 respectively.

a classification loss \mathcal{L}_{cls} , the second row represents our method with the classification loss and our proposed class associative loss, i.e., $\mathcal{L}_{cls} + \mathcal{L}_{cal}$, and the third row represents our method with the classification loss, class associative loss, and our prototype discriminative loss, i.e., $\mathcal{L}_{cls} + \mathcal{L}_{cal} + \mathcal{L}_{pdl}$. According to the results in Table 4, both the class associative loss \mathcal{L}_{cal} and prototype discriminative loss \mathcal{L}_{pdl} can improve the model's classification performance. To be specific, the class associative loss improves the classification accuracy by an average of 1.22% on six data sets, and the prototype discriminative loss \mathcal{L}_{pdl} loss also promotes the classification accuracy by an average of 0.71%. Therefore, incorporating both terms into the model is the optimal choice.

4.3.2 Parameters sensitivity. Fig. 5 shows the classification accuracy of our method CEL on benchmark data sets CIFAR-100 and CUB200 under different parameter settings. Fig. 5 (a), (b), (c) and (d) correspond to the parameters α , β , γ_1 , and γ_2 , respectively. To be specific, α and β control the weights of the class associative loss and the prototype discriminative loss, while γ_1 and γ_2 control the relative importance of the pull close and push away components within each loss. As illustrated in Fig. 5, when α is set to 0.5, β to 1, and γ_1 to 1, the model achieves the best classification performance. Specifically, when γ_2 is set to 1 and 2, our method achieves the highest classification accuracy on data sets CUB200 and CIFAR-100, respectively. Therefore, setting α , β , and γ_1 to 0.5, 1, and 1, and choosing γ_2 from {1, 2} are the suggested parameters for our method.

4.3.3 Length of the class-wise embedding. We conduct experiments on the data sets CIFAR-100 and CUB200 to verify the impact of different lengths of class-wise embedding l on model classification performance. To be specific, we select $l \in \{128, 256, 512, 768, 1024\}$. As shown in Fig. 6, for data set CIFAR-100, which has smaller image

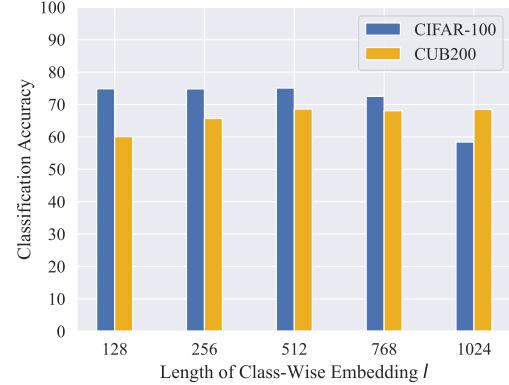


Figure 6: Classification accuracy of different lengths of class-wise embedding on data sets CIFAR-100 and CUB200.

sizes (32×32), the classification accuracy of the model is higher when the length of the class-wise embedding is less than or equal to 512. This is because excessively large embeddings can dilute important features in data sets with smaller feature dimensions, which reduces classification performance. Conversely, for the data set CUB200, which has larger image sizes (224×224), the classification accuracy is higher when the length of the class-wise embedding is greater than or equal to 512, this is because for data sets with larger feature dimensions, excessively small embeddings can compress important feature information, leading to a decline in performance. Therefore, considering both cases, setting the class-wise embedding length to 512 is a good choice.

5 CONCLUSION

In this paper, we have presented a novel method named CEL to address the IDPLL problem. For the first time, we realize that IDPLL is a mixed blessing with both positive side and negative side. We, therefore, propose to construct the class-wise embeddings to explore the relationships among the candidate labels and the non-candidate labels. To leverage the positive side of IDPLL, we introduced the class associative loss to learn representations that are more suitable for IDPLL. This is achieved by leveraging the similarity among labels within the candidate label set and the differences between the candidate label set and the non-candidate label set through class-wise embeddings. To mitigate the negative side of IDPLL, i.e., identifying the ground-truth label in the candidate set becomes more challenging, we constructed prototype discriminative loss to guide the model's disambiguation process using class prototypes which include global sample information. Extensive experiments on both common and fine-grained data sets demonstrate that our method significantly outperforms twelve state-of-the-art PLL and IDPLL methods. Moreover, compared with previous methods, our method converges faster in the early stages of model training, while produces the highest classification accuracy in the later training stages.

REFERENCES

- [1] Timothée Cour, Benjamin Sapp, and Ben Taskar. 2011. Learning from Partial Labels. *Journal of Machine Learning Research* 12 (2011), 1501–1536.
- [2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*. 248–255.
- [3] Lei Feng and Bo An. 2018. Leveraging Latent Label Distributions for Partial Label Learning. In *International Joint Conference on Artificial Intelligence*. 2107–2113.
- [4] Lei Feng and Bo An. 2019. Partial Label Learning by Semantic Difference Maximization. In *International Joint Conference on Artificial Intelligence*. 2294–2300.
- [5] Lei Feng, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama. 2020. Provably Consistent Partial-Label Learning. In *Advances in Neural Information Processing Systems* 33.
- [6] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*. 9726–9735.
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*. 770–778.
- [8] Shuo He, Guowu Yang, and Lei Feng. 2023. Candidate-aware Selective Disambiguation Based On Normalized Entropy for Instance-dependent Partial-label Learning. In *IEEE/CVF International Conference on Computer Vision*. 1792–1801.
- [9] Eytan Hüllermeier and Jürgen Beringer. 2006. Learning from ambiguously labeled examples. *Intelligent Data Analysis* 10, 5 (2006), 419–439. <http://content.iospress.com/articles/intelligent-data-analysis/ida0259>
- [10] Yuheng Jia, Xiaorui Peng, Ran Wang, and Min-Ling Zhang. 2024. Long-Tailed Partial Label Learning by Head Classifier and Tail Classifier Cooperation. In *Thirty-Eighth AAAI Conference on Artificial Intelligence*. 12857–12865.
- [11] Yuheng Jia, Fuchao Yang, and Yongqiang Dong. 2023. Partial Label Learning with Dissimilarity Propagation guided Candidate Label Shrinkage. In *Advances in Neural Information Processing Systems* 36.
- [12] Jiahao Jiang, Yuheng Jia, Hui Liu, and Junhui Hou. 2024. FairMatch: Promoting Partial Label Learning by Unlabeled Samples. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1269–1278.
- [13] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. 2011. Novel Dataset for Fine-Grained Image Categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*.
- [14] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3D Object Representations for Fine-Grained Categorization. In *2013 IEEE International Conference on Computer Vision Workshops*. 554–561.
- [15] Alex Krizhevsky. 2009. Learning Multiple Layers of Features from Tiny Images. (2009).
- [16] Jack Lanchantin, Tianlu Wang, Vicente Ordonez, and Yanjun Qi. 2021. General Multi-Label Image Classification With Transformers. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021*. 16478–16488.
- [17] Changchun Li, Ximing Li, and Jihong Ouyang. 2020. Learning with Noisy Partial Labels by Simultaneously Leveraging Global and Local Consistencies. In *ACM International Conference on Information and Knowledge Management*. 725–734.
- [18] Li-Ping Liu and Thomas G. Dietterich. 2012. A Conditional Multinomial Mixture Model for Superset Label Learning. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems*. 557–565.
- [19] Shilong Liu, Lei Zhang, Xiao Yang, Hang Su, and Jun Zhu. 2021. Query2Label: A Simple Transformer Way to Multi-Label Classification. *CoRR* abs/2107.10834 (2021). arXiv:2107.10834
- [20] Jie Luo and Francesco Orabona. 2010. Learning from Candidate Labeling Sets. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems*. 1504–1512.
- [21] Jiaqi Lv, Biao Liu, Lei Feng, Ning Xu, Miao Xu, Bo An, Gang Niu, Xin Geng, and Masashi Sugiyama. 2024. On the Robustness of Average Losses for Partial-Label Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 46, 5 (2024), 2569–2583.
- [22] Jiaqi Lv, Miao Xu, Lei Feng, Gang Niu, Xin Geng, and Masashi Sugiyama. 2020. Progressive Identification of True Labels for Partial-Label Learning. In *International Conference on Machine Learning, ICML 2020*. 6500–6510.
- [23] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. 2013. Fine-Grained Visual Classification of Aircraft. *CoRR* abs/1306.5151 (2013). arXiv:1306.5151
- [24] Congyu Qiao, Ning Xu, and Xin Geng. 2023. Decompositional Generation Process for Instance-Dependent Partial Label Learning. In *The Eleventh International Conference on Learning Representations, ICLR 2023*.
- [25] Tal Ridnik, Gilad Sharir, Avi Ben-Cohen, Emanuel Ben Baruch, and Asaf Noy. 2023. ML-Decoder: Scalable and Versatile Classification Head. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023*. 32–41.
- [26] Shiyu Tian, Hongxin Wei, Yiqun Wang, and Lei Feng. 2024. CroSel: Cross Selection of Confident Pseudo Labels for Partial-Label Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024*. 19479–19488.
- [27] Yingjie Tian, Xiaotong Yu, and Sajji Fu. 2023. Partial label learning: Taxonomy, analysis and outlook. *Neural Networks* 161 (2023), 708–734.
- [28] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. 2011. The Caltech-UCSD Birds-200-2011 Dataset. *California Institute of Technology* (2011).
- [29] Deng-Bao Wang, Min-Ling Zhang, and Li Li. 2022. Adaptive Graph Guided Disambiguation for Partial Label Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 12 (2022), 8796–8811.
- [30] Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. 2022. PiCO: Contrastive Label Disambiguation for Partial Label Learning. In *The Tenth International Conference on Learning Representations, ICLR 2022*.
- [31] Xiu-Shen Wei, Yi-Zhe Song, Oisin Mac Aodha, Jianxin Wu, Yuxin Peng, Jinhui Tang, Jian Yang, and Serge J. Belongie. 2022. Fine-Grained Image Analysis With Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 44, 12 (2022), 8927–8948.
- [32] Hongwei Wen, Jingyi Cui, Hanyuan Hang, Jiabin Liu, Yisen Wang, and Zhouchen Lin. 2021. Leveraged Weighted Loss for Partial Label Learning. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, Vol. 139. 11091–11100.
- [33] Dong-Dong Wu, Deng-Bao Wang, and Min-Ling Zhang. 2022. Revisiting Consistency Regularization for Deep Partial Label Learning. In *International Conference on Machine Learning, ICML 2022*, Vol. 162. 24212–24225.
- [34] Dong-Dong Wu, Deng-Bao Wang, and Min-Ling Zhang. 2024. Distilling Reliable Knowledge for Instance-Dependent Partial Label Learning. In *Thirty-Eighth AAAI Conference on Artificial Intelligence*. 15888–15896.
- [35] Shiyu Xia, Jiaqi Lv, Ning Xu, and Xin Geng. 2022. Ambiguity-Induced Contrastive Learning for Instance-Dependent Partial Label Learning. In *Thirty-First International Joint Conference on Artificial Intelligence*. 3615–3621.
- [36] Shiyu Xia, Jiaqi Lv, Ning Xu, Gang Niu, and Xin Geng. 2023. Towards Effective Visual Representations for Partial-Label Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023*. 15589–15598.
- [37] Ning Xu, Biao Liu, Jiaqi Lv, Congyu Qiao, and Xin Geng. 2023. Progressive Purification for Instance-Dependent Partial Label Learning. In *International Conference on Machine Learning, ICML 2023*. 38551–38565.
- [38] Ning Xu, Congyu Qiao, Xin Geng, and Min-Ling Zhang. 2021. Instance-Dependent Partial Label Learning. In *Advances in Neural Information Processing Systems 34*. 27119–27130.
- [39] Zinan Zeng, Shijie Xiao, Kui Jia, Tsung-Han Chan, Shenghua Gao, Dong Xu, and Yi Ma. 2013. Learning by Associating Ambiguously Labeled Images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 708–715.
- [40] Min-Ling Zhang and Fei Yu. 2015. Solving the Partial Label Learning Problem: An Instance-Based Approach. In *International Joint Conference on Artificial Intelligence*. 4048–4054.
- [41] Min-Ling Zhang, Bin-Bin Zhou, and Xu-Ying Liu. 2016. Partial Label Learning via Feature-Aware Disambiguation. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1335–1344.
- [42] Bo Zhao, Jiashi Feng, Xiao Wu, and Shuicheng Yan. 2017. A survey on deep learning-based fine-grained object classification and semantic segmentation. *Int. J. Autom. Comput.* 14, 2 (2017), 119–135.