# Mitigating Instance Entanglement in Instance-Dependent Partial Label Learning

## Supplementary Material

## Contents



## A. Methodological and Theoretical Details

### A.1. Theoretical Analysis of the Classification Loss

In Section 3.3, we propose a disambiguation classification loss $\mathcal{L}_{discls}$ to adjust the confidence of candidate and non-candidate labels. This section shows that $\mathcal{L}_{discls}$ can be cast as an instantiation of the Leveraged Weighted (LWS) loss family [52], and thus shares a similar Bayes-consistency intuition under the symmetric-surrogate assumption.

**Review of the LWS loss family.** For a partially labeled instance $(\boldsymbol{x}, \mathcal{S})$, a representative LWS-form loss takes the leveraged weighted form:

$$\tilde{\mathcal{L}}_{\mathrm{LWS}}(\mathcal{S}, g(\boldsymbol{x})) = \sum_{j\in\mathcal{S}} w_j\, \psi(g_j(\boldsymbol{x})) + \beta \sum_{j\notin\mathcal{S}} w_j\, \psi(-g_j(\boldsymbol{x})), \tag{1}$$

where $w_j \geq 0$ are label-wise weights, $\beta > 0$ balances the penalty on non-candidate labels, and $\psi(\cdot)$ is a non-increasing binary surrogate satisfying the symmetric condition $\psi(t) + \psi(-t) = 1$. Prior work provides a Bayes-consistency analysis for such leveraged weighted losses under symmetric surrogates and specific candidate-set generation assumptions [52].

**Formulation mapping.** Recall our proposed loss:

$$\mathcal{L}_{discls}(\boldsymbol{x}) = \sum_{j\in\mathcal{Y}} \omega_j\, \ell(s_j, \boldsymbol{x}), \tag{2}$$

where $s_j = \mathbb{I}[j \in \mathcal{S}]$ is the indicator function denoting whether label $j$ is in the candidate set $\mathcal{S}$. Under a symmetric surrogate $\psi(\cdot)$ satisfying $\psi(t) + \psi(-t) = 1$, our per-label term can be written as:

$$\ell(s_j, \boldsymbol{x}) = s_j\, \psi(g_j^q(\boldsymbol{x})) + (1 - s_j)\, \psi(-g_j^q(\boldsymbol{x})), \tag{3}$$

which yields:

$$\begin{aligned}
\mathcal{L}_{discls}(\boldsymbol{x}) &= \sum_{j\in\mathcal{Y}} \omega_j \Big[ s_j\, \psi(g_j^q(\boldsymbol{x})) + (1 - s_j)\, \psi(-g_j^q(\boldsymbol{x})) \Big] \\
&= \sum_{j\in\mathcal{S}} \omega_j \Big[ 1 \cdot \psi(g_j^q(\boldsymbol{x})) + 0 \cdot \psi(-g_j^q(\boldsymbol{x})) \Big] \\
&\quad + \sum_{j\notin\mathcal{S}} \omega_j \Big[ 0 \cdot \psi(g_j^q(\boldsymbol{x})) + 1 \cdot \psi(-g_j^q(\boldsymbol{x})) \Big] \\
&= \sum_{j\in\mathcal{S}} \omega_j \psi(g_j^q(\boldsymbol{x})) + \sum_{j\notin\mathcal{S}} \omega_j \psi(-g_j^q(\boldsymbol{x})).
\end{aligned} \tag{4}$$

Therefore, Eq. (1) reduces to our formulation by setting the leverage parameter $\beta = 1$ and matching the weights $w_j = \omega_j$. Since our $\mathcal{L}_{discls}$ admits the leveraged weighted form in Eq. (1) under a symmetric surrogate, it naturally aligns with the Bayes-consistency intuition developed for LWS losses in [52].

In practice, we optimize a cross-entropy variant for numerical stability, which preserves the same reward–penalty

**Algorithm 1** Training framework with mini-batch

**Require:** Partial label training batch $\mathcal{B}$, augmentations generation model $E$, query model $e_q$ and classifier model $g_q$ with a shared backbone, key model $e_k$ and confidence estimator model $g_k$ with a shared backbone.

**Ensure:** Optimized classifier $f(\boldsymbol{x}) = \arg\max_j g_j^q(\boldsymbol{x})$.

1: Generate class-specific augmentations $\boldsymbol{x}'_s = E(\boldsymbol{x}, I(s))$ for every $(\boldsymbol{x}, \mathcal{S}) \in \mathcal{B}$ and for each $s \in \mathcal{S}$.
2: Calculate the contrastive loss $\mathcal{L}_c(\boldsymbol{x}'_s)$ for augmentations by Eq. (3) of the main paper.
3: Calculate the disambiguation classification loss $\mathcal{L}_{discls}(\boldsymbol{x})$ for instance by Eq. (5) of the main paper.
4: Calculate the final loss $\mathcal{L}(\boldsymbol{x}, \mathcal{S})$ according to Eq. (6) and update the network $e_q$ and $g_q$.
5: momentum update $e_k$ and $g_k$ using $e_q$ and $g_q$.
6: Update the key queue and the confidence queue.

mechanism in the log-probability space. Notably, this is consistent with the empirical observations in [52], which demonstrate that the cross-entropy loss provides greater stability during the optimization process.

### A.2. Pseudo Code of Training

The complete training process is outlined in Algorithm 1. First, CAD generates class-specific augmentations for each instance based on the candidate labels and computes the contrastive loss from these augmentations (steps 1-2). Next, it calculates the disambiguation classification loss for the instances to obtain the overall loss and updates the model (steps 3-5). Meanwhile, two queues are maintained to store keys and their corresponding confidences, which are updated at each batch (step 6).

## B. Dataset Details

### B.1. Dataset Synthesis

In the main paper, following common practice [54, 56, 58], we synthesized some partial label datasets. Specifically, a model trained on the clean dataset is used as the annotator to predict the class posterior probabilities of the instances. Then, the top few labels with the highest posterior probabilities are selected as candidate labels, with the number of candidate labels controlled by a hyperparameter $\tau$. The overall algorithmic is illustrated in Algorithm 2.

### B.2. Dataset Statistics

The statistics of the datasets used in the experiments are summarized in Table 1.

**Algorithm 2** Pseudocode for data synthesis

**Require:** Dataset $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, annotator model $f : \mathcal{X} \to \mathbb{R}^C$ that maps instances sampled from feature space $\mathcal{X}$ to a $C$-dimensional class posterior space, hyperparameter $\tau$.

**Ensure:** The set of candidate labels $\mathcal{S}_i$ for each instance $x_i$ where $i \in \{1, \ldots, n\}$.

1: For all $\boldsymbol{x}_i$, estimate its class posterior probability $\boldsymbol{p}_i = f(\boldsymbol{x}_i)$, where $\boldsymbol{p}_{ij}$ represents the posterior estimate $\hat{P}(j|\boldsymbol{x}_i)$ for class $j$.
2: Compute the normalized class posterior probability as $\boldsymbol{p}'_{ij} = \frac{\boldsymbol{p}_{ij}}{\max_{j \neq y_i} \boldsymbol{p}_{ij}}$.
3: Calculate the label flip probability as $\boldsymbol{p}_{ij}^{flip} = \frac{\boldsymbol{p}'_{ij}(C-1)}{\sum_{j \neq y_i} \boldsymbol{p}'_{ij}} \times \tau$. Subsequently, if $\boldsymbol{p}_{ij}^{flip} > 1$, set $\boldsymbol{p}_{ij}^{flip} = 1$.
4: Sample from a binomial distribution with probability $\boldsymbol{p}_{ij}^{flip}$. If the result is 1, add the $j$-th label to the candidate label set $\mathcal{S}_i$ for instance $\boldsymbol{x}_i$.

Table 1. The statistics of datasets. Here, labels refers to the average number of candidate labels assigned to each instance.

| Dataset | Train | Test | Dims | Classes | Labels |
|---|---|---|---|---|---|
| Fashion-MNIST | 60,000 | 10,000 | $28 \times 28$ | 10 | 7.40 |
| CIFAR-10 | 50,000 | 10,000 | $32 \times 32$ | 10 | 5.89 |
| CIFAR-100 | 50,000 | 10,000 | $32 \times 32$ | 100 | 9.40 |
| Flower | 1,020 | 6,149 | $224 \times 224$ | 102 | 5.49 |
| Oxford-IIIT Pet | 3,680 | 3,669 | $224 \times 224$ | 37 | 3.73 |

Table 2. Statistic of entangled instances at different thresholds. #Pairs and #Instances represent the number of entangled pairs and instances, respectively. Since each instance can be entangled with multiple other instances, the number of pairs and instances may not follow a strict ratio.

| Ratio | 0.100% | | | 0.010% | | | 0.001% | | |
|---|---|---|---|---|---|---|---|---|---|
| | #Pairs | #instances | $\xi$ | #Pairs | #instances | $\xi$ | #Pairs | #instances | $\xi$ |
| Fashion-MNIST | 822,911 | 42,851 | 0.8774 | 82,291 | 13,345 | 0.9594 | 8,229 | 3,114 | 0.9859 |
| CIFAR-10 | 374,916 | 47,655 | 0.8352 | 37,491 | 17,513 | 0.9088 | 3,749 | 2,833 | 0.9478 |
| CIFAR-100 | 23,041 | 10,007 | 0.8303 | 2,304 | 1,644 | 0.8,971 | 230 | 270 | 0.9364 |
| Oxford-IIIT | 133 | 131 | 0.8972 | 13 | 22 | 0.9211 | 1 | 2 | 0.9407 |
| Flower | 5 | 8 | 0.7740 | 1 | 2 | 0.8137 | 1 | 2 | 0.8137 |

### B.3. Statistic of Entangled Instances

In this section, we provide the statistic and thresholds of entangled instance pairs with the highest similarity (top 0.1%, 0.01%, and 0.001%) in Table 2.

## C. Baselines and Implementation

### C.1. Details of Baselines

We compared our method with the advanced PLL methods including POP [59], VALEN [58], IDGP [41], CP-DPLL [53], CAVL [64], PICO [50], PRODEN [34], LWS [52], RC [13], CC [13], ABLE [56], and DIRK [54]. These

methods represent a range of strategies, including label refinement, variational inference, knowledge distillation, and contrastive learning. We compared our method with the following advanced PLL methods: POP [59]: a label refinement algorithm that iteratively updates both the classifier and the labels; VALEN [58]: an algorithm that uses variational inference to estimate the latent label distribution; IDGP [41]: a framework utilizes separate categorical and Bernoulli models to capture instance-specific label dependencies and maximizes posterior probabilities; CP-DPLL [53]: a framework that integrates supervised learning on non-candidate labels with consistency regularization on candidates; CAVL [64]: a label disambiguation algorithm that adopts the class activation map mechanism for label selection to obtain potentially reliable labels; PICO [50]: a framework that uses contrastive learning to help label disambiguation; PRODEN [34]: a self-training framework that incrementally refines label assignments; LWS [52]: a loss weighting strategy that balances the losses of candidate labels and non-candidate labels; RC [13]: a label weighting algorithm designed to construct risk-consistent loss; CC [13]: a label weighting algorithm designed to construct classifier-consistent loss; CEL [61]: a class-wise embedding framework that uses associative and prototype losses to explore intra-sample relationships; ABLE [56]: a framework that leverages label ambiguity to construct a contrastive learning setup, aiding in representation learning for PLL; DIRK [54]: an algorithm that learns reliable label distributions by rectifying the relative confidence of candidate and non-candidate labels.

### C.2. Additional Implementation Details

This section provides further implementation details that complement the descriptions in the main paper. All experiments in this paper are implemented using the PyTorch framework. For datasets including Fashion-MNIST, CIFAR-10, CIFAR-100, and PLCIFAR10, training was conducted on a single NVIDIA RTX 4090 GPU. For Oxford Flowers and Oxford-IIIT Pet, we used a single NVIDIA A100 GPU due to their higher resolution and increased memory requirements. To ensure fair comparison, all baseline methods are implemented using a unified backbone and the same data augmentation pipelines [54]. For hyperparameters of CAD, we empirically set default $\beta = 1$, $\tau = 0.12$, and $\tau_2 = 0.4$ following [54]. Below, we provide additional details on the implementation of the CAD-CAM method and the data augmentation strategies used in our framework.

### C.3. Implementation Details of CAD-CAM

To implement the CAM-based augmentation required by CAD-CAM, we first pretrain the model for 50 epochs using a weighted PLL cross-entropy loss. After this warm-up phase, we apply the proposed augmentation strategy: for each candidate label in a sample, we compute the corresponding class activation map (CAM) and generate class-specific augmented samples accordingly. Specifically, we compute standard CAMs as a channel-weighted sum of feature maps followed by ReLU, then normalize and resize them to the input resolution; we retain the top 30% activations to form class-specific masks. Once the augmented samples are obtained, the contrastive loss is introduced into the training objective. These class-specific augmentations are refreshed every 50 epochs using the updated model predictions. If a CAM is nearly uniform—i.e., no regions are strongly activated—the corresponding class-specific augmentation is discarded for that class. To mitigate potential artifacts or sharp transitions introduced during this augmentation process, we further apply a Gaussian blur with a kernel size of 5 and a standard deviation of 1.0 to each augmented image.

In our implementation, we empirically set the contrastive loss weight to $\beta = 2.0$ and the temperature parameter to $\tau_2 = 0.1$. And to mitigate potential artifacts or sharp transitions introduced during the augmentation process, we apply a Gaussian blur transformation with a kernel size of 5 and a standard deviation of 1.0 to each augmented image. All other hyperparameters are kept consistent with the original CAD setting.

### C.4. Data Augmentation Details

We adopt three data augmentation strategies in our framework: a key view and a query view for contrastive learning, and a disambiguation view for label disambiguation. For class-specific augmentation, we employ both CAM-based and class-specific diffusion–based methods. Following DIRK [54], class-specific augmented samples are further transformed to generate the key and query views, which are used to compute the contrastive loss. The original samples are augmented into disambiguation views for computing the PLL classification loss with confidence adjustment. For CIFAR-10, PLCIFAR10, and CIFAR-100, the key view is generated via random flipping, resizing, cropping, color jittering, and random grayscaling; the query view uses flipping, resizing, cropping, and RandAugment [10]; and the disambiguation view is created using flipping, cropping, padding, Cutout [11], and AutoAugment [9]. For Fashion-MNIST, the disambiguation view includes horizontal flipping, reflective cropping, Cutout, and grayscale conversion; the key view uses resized cropping and flipping; and the query view further includes ColorJitter and random grayscaling. For Oxford Flowers and Pets, we apply RandomResizedCrop, flipping, and ImageNetPolicy for the disambiguation view; the key view incorporates ColorJitter and grayscale; and the query view adopts RandomAugment. To ensure fair comparison, we apply the same augmenta-

tion pipeline across all methods. Moreover, all baselines are implemented within a unified knowledge distillation framework following DIRK.

# D. Additional Experiments

## D.1. Comparison of Training Cost

While our main instantiation of CAD leverages a pre-trained diffusion model for class-specific augmentation, this component belongs to the framework rather than the core objective and can be replaced by lighter alternatives. On CIFAR-10, diffusion-based editing takes 5.91 GPU-hours on a single RTX 4090, accounting for about 24% of CAD's total cost of 24.66 GPU-hours (Table 3). Since the diffusion stage is performed offline and is highly parallelizable, its wall-clock time can be further reduced by distributing it across multiple GPUs. Moreover, when efficiency is a primary concern, CAD can be instantiated with the CAM-based variant CAD-CAM, which avoids diffusion entirely and reduces the total cost to 18.50 GPU-hours on CIFAR-10 and 25.23 GPU-hours on CIFAR-100.

Compared with existing ID-PLL methods, the additional GPU-hours of CAD and CAD-CAM mainly arise from the increased number of class-specific augmentations participating in training. The training costs of several representative methods are summarized in Table 3. Despite this extra GPU-time overhead, CAD and its lightweight variant CAD-CAM achieve substantial performance improvements over these baselines.

Table 3. GPU-hours on CIFAR-10 and CIFAR-100 benchmarks.

| Method | CIFAR-10 | CIFAR-100 |
|---|---|---|
| DIRK | 10.19 | 14.37 |
| ABLE | 13.89 | 19.15 |
| CAD-CAM | 18.50 | 25.23 |
| CAD | 24.66 | 35.54 |

## D.2. Experiments with Diffusion-based Augmentation

This section compares the performance of DIRK, ABLE, and RC with their augmentations using the same diffusion-augmented data as CAD (denoted as DIRK-C, ABLE-C, and RC-C) on the Fashion-MNIST, CIFAR-10, and CIFAR-100 benchmarks. All methods are evaluated under identical experimental conditions: they use the same original training data combined with the same class-specific diffusion-augmented data, employ identical input transformations, and are implemented within the same backbone.

As shown in Table 4, the accuracy of other baseline methods decreases after incorporating the augmentation data. This supports the analysis in Section E.1, where directly using such data with the label induced them introduces more noise, ultimately affecting classification performance. **It is worth noting that this is why these data were not used for other baseline comparisons in Table 2**, despite it seemingly being a fairer approach. In contrast, CAD effectively extracts useful signals by aligning same-class enhanced features, which is plug-and-play and consistently boosts baselines (Section D.3).

## D.3. Plug-and-Play Integration with Existing Methods

To evaluate the generality of our approach, we plug CAD into several representative ID-PLL methods, including POP, ABLE, and DIRK. As shown in Table 5, CAD consistently improves all baselines across both CIFAR-10 and CIFAR-100. Notably, even strong methods such as DIRK benefit from CAD, achieving gains of 1.99% and 3.58% on CIFAR-10 and CIFAR-100, respectively. These results demonstrate that CAD serves as a flexible and effective plug-and-play module that can be seamlessly integrated into existing frameworks to enhance instance disentanglement and boost classification performance.

## D.4. Experiments on PLCIFAR10

To evaluate the real-world applicability of CAD, we additionally tested it on the PLCIFAR10 dataset, which contains partial labels derived from human annotations. As shown in Table 6, CAD outperforms the competitive baselines DIRK and ABLE. These results demonstrate CAD's ability to generalize to real-world scenarios.

## D.5. Experiments for Confusion Matrix

**Confusion matrix for Fashion-MNIST.** As a complementary for confusion matrix experiments in the main paper, in Figure 2, we further provide the confusion matrix for Fashion-MNIST benchmarks. And in Figure 3, we provide the ablation results for confusion matrix in Fashion-MNIST, results shows that both stretegis we used are contributed for class confusion.

**Label overlap for CIFAR-10 and Fashion-MNIST.** To better observe which classes are more prone to label confusion, Figure 4 shows the label overlap matrices for CIFAR-10 and Fashion-MNIST, which are datasets with fewer classes for easier visualization. The element in the $i$-th row and $j$-th column of the matrix represents the proportion of instances in classes $i$ and $j$ that share both the $i$ and $j$ labels. Combined with the experimental results from the confusion matrix, we can see that most of the classes with higher label overlap are also the ones that are harder to distinguish in the confusion matrix, and our method shows significant improvement primarily on these classes.

Table 4. Comparison of accuracy (expressed in percentage) with Diffusion-based Augmentation. The best results for each benchmark are highlighted in **bold**.

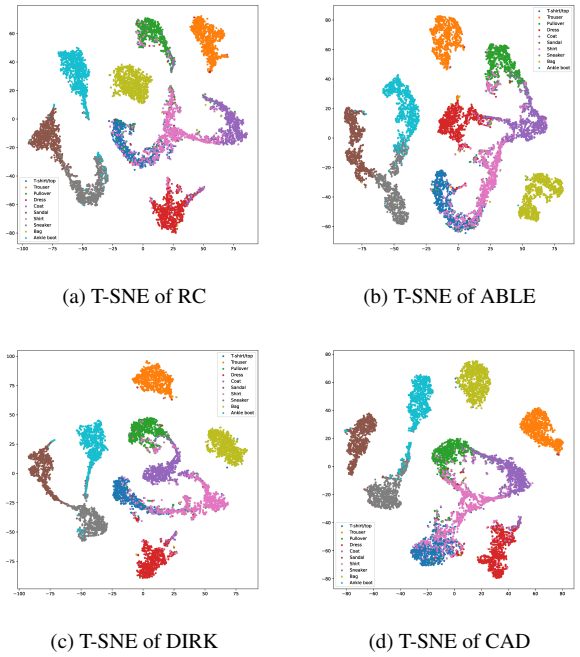
| | Fashion-MNIST | CIFAR-10 | CIFAR-100 |
|---|---|---|---|
| RC-C | 66.66 ± 0.99 | 76.80 ± 0.27 | 59.63 ± 0.20 |
| RC | 84.87 ± 1.48 | 87.53 ± 0.94 | 65.26 ± 0.40 |
| ABLE-C | 78.41 ± 0.35 | 59.66 ± 1.04 | 58.03 ± 0.28 |
| ABLE | 89.81 ± 0.08 | 83.92 ± 0.67 | 63.92 ± 0.39 |
| DIRK-C | 79.05 ± 0.49 | 85.06 ± 0.15 | 62.66 ± 0.18 |
| DIRK | 91.48 ± 0.21 | 91.48 ± 0.21 | 68.77 ± 0.49 |
| CAD | **92.14 ± 0.91** | **93.57 ± 0.24** | **72.03 ± 1.88** |

Table 5. Accuracy (%) comparison on CIFAR-10 and CIFAR-100. "+CAD" denotes equipping the baseline with our proposed CAD contrastive learning module.

| Method | CIFAR-10 | CIFAR-100 |
|---|---|---|
| POP | 89.55 ± 0.36 | 64.57 ± 0.37 |
| POP+CAD | **91.56 ± 0.80** | **66.56 ± 2.86** |
| ABLE | 83.92 ± 0.67 | 63.92 ± 0.39 |
| ABLE+CAD | **85.21 ± 4.08** | **68.63 ± 1.13** |
| DIRK | 90.87 ± 0.25 | 68.77 ± 0.49 |
| DIRK+CAD | **92.86 ± 0.17** | **72.35 ± 0.53** |

Table 6. Accuracy (%) on the PLCIFAR10 dataset with partial labels from human annotations.

| Method | Accuracy (%) |
|---|---|
| ABLE | 91.82 |
| DIRK | 92.91 |
| CAD (Ours) | **93.24** |



(a) T-SNE of RC (b) T-SNE of ABLE

(c) T-SNE of DIRK (d) T-SNE of CAD

Figure 1. T-SNE visualization on the Fashion-MNIST benchmark, with different colors representing different classes.

## D.6. Parameter Sensitivity Analysis

The main hyper-parameter introduced in this method is the weight $\beta$. In the main experiments, we naively set it to 1. To analyze its impact, we provide the results with different $\beta$ on the CIFAR-10 dataset in Table 7. The results show that the method is generally robust to parameter changes, with the best performance observed when $\beta$ is set in the range of $[1.00, 1.75]$. This suggests that balancing representation learning and label disambiguation in the ID-PLL task helps improve model performance.

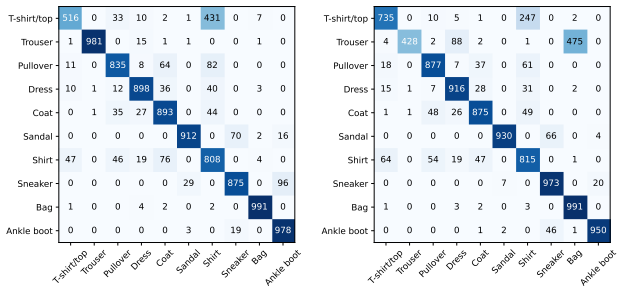
Additionally, we conducted a sensitivity analysis on CIFAR-10 to assess three key contrastive learning hyper-parameters:

- Contrastive weight $\beta \in \{0.0, 0.5, 1.0\text{(default)}, \mathbf{1.5}, 2.0\}$ , with accuracy: 91.21%, 93.20%, 93.57%, **94.10%**, 93.64%;
- Temperature $\tau \in \{0.03,\ 0.05,\ 0.07\ \text{(default)},\ 0.09,\ \mathbf{0.12},\ 0.15,\ 0.18\}$, with accuracy: 92.64%, 93.48%, 93.57%, 93.70%, **94.23%**, 93.71%, 93.68%;
- Momentum temperature $\tau_2 \in \{\mathbf{0.1}, 0.4\text{(default)}, 0.7\}$, with accuracy: **94.58%**, 93.57%, 93.51%

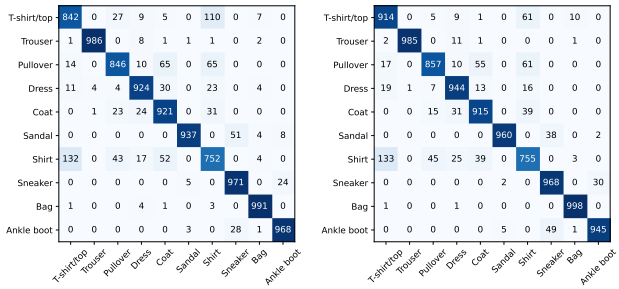
The default hyperparameters were inherited from prior works. Results show CAD is robust to a wide range of settings, and moderate tuning (e.g., $\beta = 1.5$, $\tau = 0.12$, $\tau_2 = 0.1$) can further improve performance.

Table 7. Accuracy with different $\beta$ in the CIFAR-10 benchmark. The best results are highlighted in **bold**.

| $\beta$ | 0.00 | 0.25 | 0.50 | 0.75 | 1.00 | 1.25 | 1.50 | 1.75 | 2.00 |
|---|---|---|---|---|---|---|---|---|---|
| Accuracy | 89.19 | 92.53 | 92.95 | 92.94 | 93.85 | **94.10** | 93.97 | 93.88 | 92.87 |

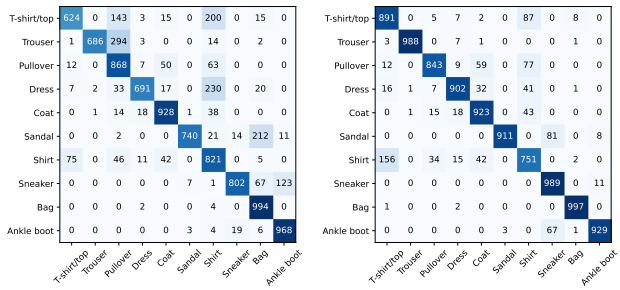


(a) Confusion Matrix for RC (b) Confusion Matrix for ABLE

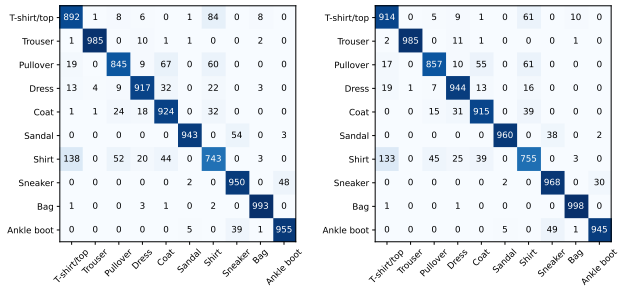


(c) Confusion Matrix for DIRK (d) Confusion Matrix for CAD

Figure 2. Confusion Matrix on the Fashion-MNIST benchmark.

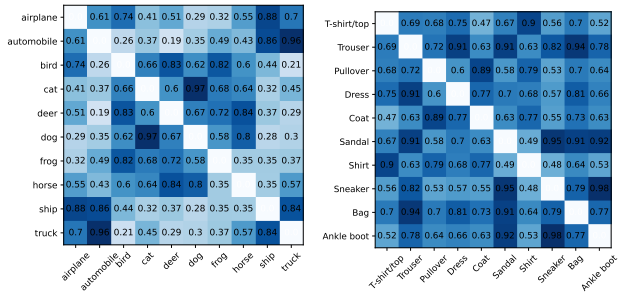


(a) Without Both (b) Without LD



(c) Without RL (d) CAD

Figure 3. Ablation of Confusion Matrix on the Fashion-MNIST benchmark.



(a) CIFAR-10 (b) Fashion-MNIST

Figure 4. Label Overlap for the CIFAR-10 and Fashion-MNIST benchmarks.

## D.7. Experiments on Low-entanglement Conditions

To study CAD under low-entanglement conditions, we removed the annotator's most confusing class during candidate construction. In this case, CAD achieves 95.60% on CIFAR-10, comparable to DIRK (95.65%) and ABLE (95.31%). These results support the paper's claim that instance entanglement is a key challenge in partial-label learning. When entanglement is reduced, label ambiguity decreases, and all methods tend to converge-yet CAD remains competitive, confirming its robustness across varying entanglement levels.

## D.8. Experiments on text classification

To evaluate the generalization of CAD beyond vision, we applied CAD to the AGNews text classification task using BERT-base as the backbone encoder. For class-specific data augmentation, we employed T5-base [42] to generate synthetic samples via prompt-based rewriting. Specifically, for each original news article labeled with class $c$ (e.g., "World", "Sports", "Business", or "Sci/Tech"), we used one of the following class-conditional prompts:

- Rewrite this article with richer {class} terms:
- Restyle the following report using {class} language:
- Retell the following article using {class} phrases:

where {class} is replaced by the name of the corresponding augmented class. The generated texts are then used as augmented training samples within CAD's feature alignment framework. As shown in Table 8, CAD outperforms both ABLE and DIRK, demonstrating its generalization capability to non-vision domains such as text classification.

Table 8. Text classification accuracy (%) on AGNews. All methods use BERT-base as the student encoder; augmented data are generated using T5-base with class-specific prompts.

| Method | AGNews Accuracy (%) |
|---|---|
| DIRK | 77.34 |
| ABLE | 93.92 |
| CAD (Ours) | **94.30** |

### D.9. Recovered Rate under Full Supervision

We report the recovery rate, i.e., the fraction of entangled instances misclassified but correctly classified under full supervision. Table 9 shows the recovery rate for a representative baseline (DIRK), showing values up to 79.43% (CIFAR-100, $\xi = 0.9$), indicating these samples are learnable under full supervision but hindered by entanglement.

Table 9. Recovery rate (%) of entangled instances misclassified by DIRK but correctly classified under full supervision (ground-truth labels), at different similarity thresholds $\xi$.

| $\xi$ | F-MNIST | CIFAR-10 | CIFAR-100 |
|---|---|---|---|
| 0.90 | 16.01 | 27.45 | 79.43 |
| 0.95 | 15.28 | 26.56 | 51.06 |
| 0.99 | 11.51 | 25.00 | 38.46 |

## E. Visualization Analysis

### E.1. Augmentation Analysis

This section presents a visualization analysis of augmentations produced by diffusion-based and CAM-based methods, with the aim of assessing whether each method effectively emphasizes class-discriminative regions.

**Analysis for diffusion-based augmentation.** To intuitively analyze whether the generated augmentations are helpful, we present some of the visualization results in CIFAR-10 benchmarks. Figure 5 shows four sets of examples: a frog instance labeled as {cat, dog, frog}, a truck instance labeled as {truck, automobile, horse, frog, deer, ship}, a deer instance labeled as {bird, deer, dog, frog, horse, truck}, and an automobile instance labeled as {airplane, automobile, cat, dog, ship, truck}. It can be observed that the augmentations generated by guiding instances with their ground-truth labels (marked in blue) generally do not show significant changes. This helps preserve the original information of the instance in contrastive learning. For other class-specific augmentations, we can see that most of them exhibit some characteristics of the corresponding class while preserving the original instance's color and texture. Maximizing the preservation of instance-specific characteristics while enhancing classes-specific characteristics can help the model better distinguish between class-related features and the instance's unique traits.

However, some of these augmentations exhibit noticeable defects (marked in red): while class-related features are successfully amplified—such as the oceanic blue tones and reflective gloss in the ship-like example (second row)—the edits often introduce artifacts or semantic inconsistencies. Directly adding such diffusion-edited instances to existing methods and treating the induced labels as supervision degrades performance, largely due to these imperfections and the resulting distribution shift. This indicates that simply injecting edited samples is insufficient; instead, their utility depends on how they are structurally integrated into the learning objective. Therefore, rather than using these imperfect augmentations directly for classifier training, we employ them in representation learning with quality-based reweighting, which allows us to harness their amplified class signals while mitigating the negative impact of their imperfections.

Furthermore, to quantify semantic alignment, we have computed the CLIP-score margin $m(\boldsymbol{x}') = \text{sim}(\boldsymbol{x}', t) - \text{sim}(\boldsymbol{x}', y)$ for a generated sample $\boldsymbol{x}'$, comparing its similarity to the target class $t$ versus the original class $y$. As shown in Table 10, margins flip from negative to positive with a +17.21% avg. in target CLIP-score, confirming improved alignment with the target semantics.

**Analysis for CAM-based augmentation.** Figure 6 presents several examples of CAM-based augmentations generated from CIFAR-10. Here, we show the augmented samples actually used during training—specifically, those for which a salient activation region exists (as described in Appendix C.3, samples with nearly uniform activation maps are discarded).

Despite lacking explicit semantic editing capabilities, the CAM-based strategy effectively amplifies class-relevant features by suppressing non-discriminative regions. For instance, in the first row, when the guiding label is deer, the augmentation preserves the deer's head and body; when guided by the label dog, only parts resembling a dog's torso and limbs are retained. Similarly, for the truck example in the fourth row, using truck as the guide retains most of the cab, whereas the automobile label leads to preservation of only the lower part of the vehicle along with some metallic background structures. When guided by airplane, the augmentation highlights a side panel of the trailer whose shape resembles an aircraft wing.

After obtaining these class-specific augmentations, we align only augmentation pairs that share the same guiding label. Because each view retains solely the regions activated by that label, the resulting contrastive alignment

Table 10. Comparison of CLIP-score margin and target similarity (margin | CLIP-score).

| Method | Fashion-MNIST | CIFAR-10 | CIFAR-100 | Flower | Oxford-IIIT Pet |
|---|---|---|---|---|---|
| Original ($x$) | -0.0276 \| 0.2364 | -0.0415 \| 0.2151 | -0.0416 \| 0.2176 | -0.0010 \| 0.2185 | -0.0002 \| 0.2282 |
| Diffusion ($x'$) | **0.0120** \| **0.2555** | **0.0314** \| **0.2566** | **0.0181** \| **0.2453** | **0.0243** \| **0.2784** | **0.0169** \| **0.2705** |

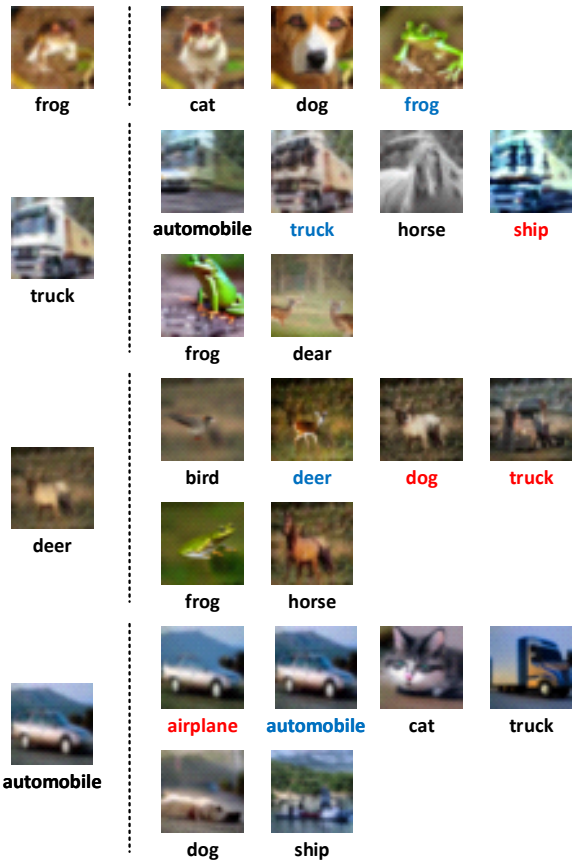



Figure 5. Diffusion-based visualization of the class-specific augmentations generated from CIFAR-10. The left side shows the samples with their ground truth labels, while the right side displays the augmentations generated with different candidate label guidance.

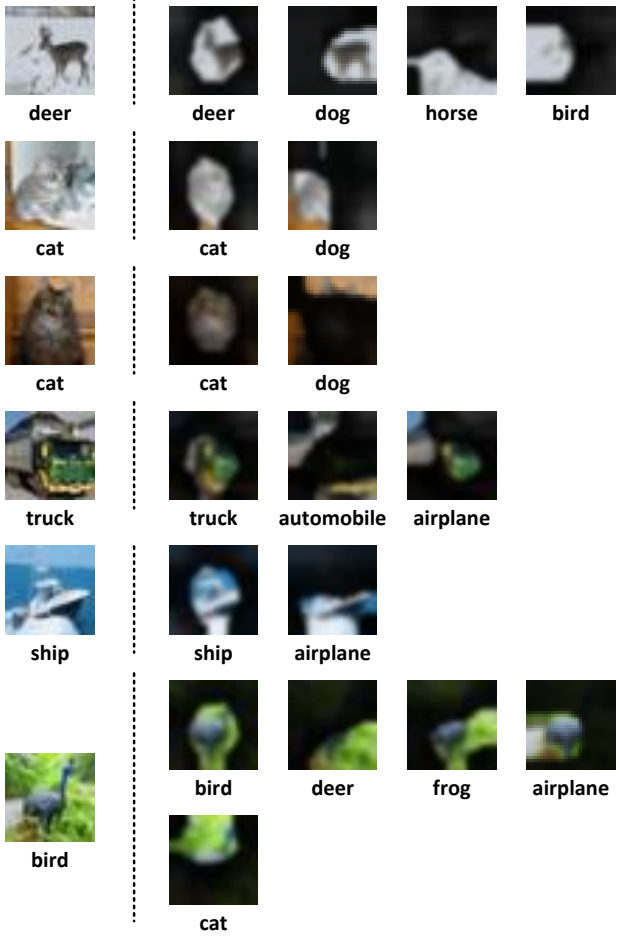



Figure 6. CAM-based visualization of the class-specific augmentations generated from CIFAR-10. The left side shows the samples with their ground truth labels, while the right side displays the augmentations generated with different candidate label guidance.

operates between semantically consistent local features, thereby avoiding the cross-class mismatches typical of direct instance-level alignment.

## E.2. T-SNE for Fashion-MNIST

In this section, we provide the T-SNE visualization for Fashion-MNIST as a complement to the experiments in Figure 4 of the main paper. The results show that our proposed method has clearer class boundaries. A notable example is the clusters of blue, brown, and gray colors on the left. In CAD (Figure 1d), although there are some errors, the boundaries of the three clusters are well-separated. In contrast, while DIRK (Figure 1c) achieves similar accuracy to CAD, the boundary samples of the three clusters are mixed together. This demonstrates that CAD effectively encourages clearer class boundaries. Overall, RC exhibits the most color overlap, which is due to its significantly lower accuracy compared to the other three methods. While the color overlap in ABLE and DIRK is less than that in RC, the confusion of boundary samples has not improved. In fact, on Fashion-MNIST, it is even worse than RC. This is because both ABLE and DIRK use candidate labels to guide contrastive learning, leading to negative effects from the misalignment of samples near similar class boundaries. In contrast, CAD shows the clearest boundaries on both bench-

marks, demonstrating that the proposed strategy effectively manages class boundaries and reduces class confusion.

# F. Additional Related Work

## F.1. Partial Label Learning

Partial Label Learning, also known as Ambiguous Label learning [4, 23, 28] or Superset Label learning [32, 33], is a weakly supervised learning task where each training instance is annotated with a set of candidate labels that includes the ground truth. Early PLL research used a straightforward average-based strategy that treated each candidate label equally [8, 23]. However, the ambiguity in the labels could mislead the model's training. To address this issue, identification-based strategies, which explicitly handle ambiguous labels, have received attention [24, 32, 63]. These studies have introduced numerous learning techniques to aid in the disambiguation, such as reweighting [13, 45, 52], employing k-nearest neighbors [23, 66], maximizing margins [36, 63], self-training [46], graph learning [35, 49], meta-learning [57], and contrastive learning [50]. Most of these methods assume that candidate labels are sampled independently of the instance, which hinders their application in real-world scenarios.

Recently, many studies have explored instance-dependent PLL tasks [18, 41, 54, 56, 58, 59]. For example, VALEN [58] restores the underlying label distribution by inferring the posterior density using Dirichlet density. ABLE [56] leverages instance-dependent characteristics to assist label disambiguation through ambiguity-guided contrastive learning. DIRK [54] constructs reliable pseudo-labels by controlling the confidence of negative labels. However, we note that these works lack explicit governance of similar class confusion, which is a common problem in ID-PLL. It should be noted that although the early maximum margin strategies can implicitly control the distance between similar classes [36, 63], these methods face challenges in scalability to large-scale high-dimensional data, while lacking entanglement handling from both representation learning and confidence adjustment.

## F.2. Contrastive Learning

Contrastive learning is a representation learning technique that works by bringing similar instances closer together and pushing dissimilar instances farther apart in the representation space. It has demonstrated its effectiveness across a wide range of supervised and unsupervised representation learning tasks [6, 7, 17, 25, 37]. Building on this success, contrastive learning has naturally been applied to many weakly supervised learning tasks, such as complementary label learning [43], noisy label learning [15, 30], and semi-supervised learning [44]. A crucial aspect of these methods is identifying pairs of instances from the same class (referred to as positive pairs) and guiding them towards aligned representations, which integrates contrastive learning with supervised information. In recent years, some methods [50, 54, 56] have explored adapting supervised contrastive learning to PLL. These methods often rely on pseudo-labels or feature distances based on model outputs to find the positive pairs. However, in ID-PLL, instances from similar classes often share the same labels, making them prone to mistaking them as false positive pairs. In this paper, unlike previous approaches, we treat class-specific augmentations bootstrapped by the same label as positive pairs to mitigate such false positives.

## F.3. Diffusion-based Image Editing

Diffusion models, inspired by principles of non-equilibrium thermodynamics, generate data matching the original distribution by progressively adding noise to the data and learning the reverse process [21]. In recent years, diffusion models have gained particular interest in image editing due to their greater control compared to the GAN-based methods. Depending on the type of editing, these studies can be categorized into semantic editing [2, 3, 14, 16, 22, 26, 27, 38, 40, 60, 67, 68], stylistic editing [1, 3, 14, 19, 22, 26, 47, 48, 51, 55], and structural editing [12, 12, 29, 39]. Instructional editing [3, 14, 16, 31, 65] is a key direction in the semantic editing domain, enabling users to guide the image editing process through natural language instructions. For example, InstructPix2Pix [3] first achieves image content editing by constructing an instruction-image paired dataset. Building on this, MoEController [31] and FOI [16] enhance editing accuracy and efficiency by introducing expert modules and improving implicit alignment capabilities, respectively. Recently, methods like MagicBrush [65] and InstructDiffusion [14] have further improved image editing performance and diversity through more comprehensive dataset construction and enhancement strategies. MagicBrush enriches its dataset by employing human editing sessions, while InstructDiffusion expands its data using tool-generated content and Photoshop requests sourced from the internet. Notably, the concept of diffusion has recently gained attention in some weakly- or semi-supervised tasks [5, 20, 62].

# G. Prompt with Cues for Oxford-IIIT Pet

- Abyssinian: sleek medium-sized cat with short dense ticked reddish-brown coat, almond-shaped green or gold eyes, large pointed ears, slender muscular body, no stripes or spots.
- American Bulldog: large muscular dog with broad head, strong jaws, short white or white-with-patches coat, loose facial skin, powerful stance.
- American Pit Bull Terrier: medium-sized athletic dog with short glossy coat in various colors, broad head,

strong neck, defined cheek muscles, short tail, alert agile build.

- Basset Hound: low-slung heavy-boned dog with extremely long ears, droopy eyes, loose skin, short legs, tricolor (black-white-tan) coat, long curved tail.
- Beagle: small to medium hound with short tricolor (black-white-tan) coat, large floppy ears, expressive brown eyes, compact muscular body, white-tipped upright tail.
- Bengal: medium to large cat with short sleek spotted or marbled golden/brown/silver coat, muscular build, green or gold eyes, wild leopard-like pattern.
- Birman: medium-sized cat with semi-long silky white or cream coat, colorpoint markings, deep blue eyes, distinctive white 'gloves' on all four paws.
- Bombay: medium-sized cat with short jet-black glossy coat, copper or gold eyes, rounded head, sleek muscular body, panther-like appearance.
- Boxer: medium-large square-built dog with short fawn or brindle coat, black facial mask, undershot jaw, docked tail, alert energetic expression.
- British Shorthair: stocky round cat with dense short blue-gray or solid-color coat, large round copper eyes, broad face with full cheeks, plush teddy-bear look.
- Chihuahua: tiny dog with large erect ears, apple-shaped head, smooth or long coat in various colors, prominent eyes, often tail curled over back.
- Egyptian Mau: medium cat with short spotted silver/bronze/smoke coat, green gooseberry eyes, 'M'-shaped forehead marking, athletic graceful build.
- English Cocker Spaniel: medium sporting dog with long silky feathered coat (parti-color or solid), long pendulous ears, soft expressive eyes, docked tail.
- English Setter: elegant medium-large gundog with long feathered white coat speckled in black/orange/lemon ('belton'), long ears, gentle refined face.
- German Shorthaired Pointer: lean athletic hunting dog with short liver-and-white or solid liver coat, docked tail, long muzzle, muscular build, alert intelligent look.
- Great Pyrenees: very large majestic dog with thick pure white double coat, dark eyes, often double dewclaws on hind legs, calm bear-like appearance.
- Havanese: small toy dog with long silky wavy or curly coat in various colors, plumed tail over back, dark eyes, lively cheerful expression.
- Japanese Chin: tiny flat-faced toy dog with long silky black-and-white or red-and-white coat, large wide-set eyes, short muzzle, feathered arched tail.
- Keeshond: medium spitz with thick gray-black-cream double coat, facial 'spectacles' markings, curled tail, foxy expression, abundant neck ruff.
- Leonberger: giant dog with lion-like mane (males), long water-resistant golden-yellow to red-brown coat, black mask, webbed feet, gentle giant look.
- Maine Coon: large rugged cat with long shaggy coat, tufted ears and paws, bushy tail, rectangular body, lynx-like ear tips, friendly expression.
- Miniature Pinscher: small compact dog with short smooth red/black-and-tan/chocolate coat, erect ears, docked tail, high-stepping gait, alert fearless look.
- Newfoundland: massive heavy-boned dog with thick water-resistant black/brown/white-and-black double coat, webbed feet, droopy jowls, sweet calm expression.
- Persian: round flat-faced cat with extremely long thick coat, small ears, large round copper eyes, cobby body, doll-like face.
- Pomeranian: tiny spitz with abundant fluffy double coat (often orange), fox-like face, small erect ears, plumed tail curled over back.
- Pug: small square dog with wrinkled face, short muzzle, curled tail, smooth fawn or black coat, large dark eyes, compact muscular body.
- Ragdoll: large semi-longhaired cat with soft plush coat, blue eyes, colorpoint pattern (seal/blue/etc.), floppy when held, gentle expression.
- Russian Blue: medium cat with short dense silvery-blue double coat, green eyes, wedge-shaped head, elegant fine-boned build, graceful posture.
- Saint Bernard: enormous dog with thick red-and-white or mahogany-and-white coat, massive head with droopy jowls, calm benevolent expression, short or long coat variant.
- Samoyed: medium-large spitz with thick pure white double coat, black lips/nose, 'smiling' expression, curled tail, erect triangular ears.
- Scottish Terrier: small sturdy terrier with wiry black (or brindle/wheaten) coat, long beard and eyebrows, short legs, dignified independent look.
- Shiba Inu: small fox-like dog with short red/sesame/black-and-tan coat, curled tail, triangular erect ears, bold alert expression, compact agile build.
- Siamese: sleek slender cat with short cream coat and dark colorpoint ears/face/paws/tail, almond-shaped blue eyes, wedge-shaped head, elegant posture.
- Sphynx: hairless cat with wrinkled skin, large lemon-shaped eyes, prominent cheekbones, large ears, muscular body, may have fine downy fuzz.
- Staffordshire Bull Terrier: compact muscular dog with short smooth coat in various colors, broad head with pronounced cheek muscles, dark eyes, short tail, affectionate look.
- Wheaten Terrier: medium terrier with soft wavy wheaten (golden-beige) coat, dark eyes, square head, topknot of longer hair, cheerful expression.
- Yorkshire Terrier: tiny toy dog with long silky steel-blue

and tan coat, small V-shaped ears, compact body, tail held high, elegant feisty look."

## References


[1] Rumeysa Bodur, Erhan Gundogdu, Binod Bhattarai, Tae-Kyun Kim, Michael Donoser, and Loris Bazzani. iedit: Localised text-guided image editing with weak supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7426–7435, 2024. 9

[2] Manuel Brack, Felix Friedrich, Katharia Kornmeier, Linoy Tsaban, Patrick Schramowski, Kristian Kersting, and Apolinário Passos. Ledits++: Limitless image editing using text-to-image models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8861–8870, 2024. 9

[3] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023. 9

[4] Ching-Hui Chen, Vishal M Patel, and Rama Chellappa. Learning from ambiguously labeled face images. *IEEE transactions on pattern analysis and machine intelligence*, 40(7):1653–1667, 2017. 9

[5] Jian Chen, Ruiyi Zhang, Tong Yu, Rohan Sharma, Zhiqiang Xu, Tong Sun, and Changyou Chen. Label-retrieval-augmented diffusion models for learning from noisy labels. *Advances in Neural Information Processing Systems*, 36, 2024. 9

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 9

[7] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 9

[8] Timothee Cour, Ben Sapp, and Ben Taskar. Learning from partial labels. *The Journal of Machine Learning Research*, 12:1501–1536, 2011. 9

[9] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 3

[10] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 3

[11] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3

[12] Dave Epstein, Allan Jabri, Ben Poole, Alexei Efros, and Aleksander Holynski. Diffusion self-guidance for controllable image generation. *Advances in Neural Information Processing Systems*, 36:16222–16239, 2023. 9

[13] 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, 3, 9

[14] Zigang Geng, Binxin Yang, Tiankai Hang, Chen Li, Shuyang Gu, Ting Zhang, Jianmin Bao, Zheng Zhang, Houqiang Li, Han Hu, et al. Instructdiffusion: A generalist modeling interface for vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12709–12720, 2024. 9

[15] Aritra Ghosh and Andrew Lan. Contrastive learning improves model robustness under label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2703–2708, 2021. 9

[16] Qin Guo and Tianwei Lin. Focus on your instruction: Fine-grained and multi-instruction image editing by attention modulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6986–6996, 2024. 9

[17] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 9

[18] Shuo He, Guowu Yang, and Lei Feng. Candidate-aware selective disambiguation based on normalized entropy for instance-dependent partial-label learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1792–1801, 2023. 9

[19] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 9

[20] Cheng-Ju Ho, Chen-Hsuan Tai, Yen-Yu Lin, Ming-Hsuan Yang, and Yi-Hsuan Tsai. Diffusion-ss3d: Diffusion model for semi-supervised 3d object detection. *Advances in Neural Information Processing Systems*, 36:49100–49112, 2023. 9

[21] Yi Huang, Jiancheng Huang, Yifan Liu, Mingfu Yan, Jiaxi Lv, Jianzhuang Liu, Wei Xiong, He Zhang, Shifeng Chen, and Liangliang Cao. Diffusion model-based image editing: A survey. *arXiv preprint arXiv:2402.17525*, 2024. 9

[22] Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly ddpm noise space: Inversion and manipulations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12469–12478, 2024. 9

[23] Eyke Hüllermeier and Jürgen Beringer. Learning from ambiguously labeled examples. *Intelligent Data Analysis*, 10 (5):419–439, 2006. 9

[24] Rong Jin and Zoubin Ghahramani. Learning with multiple labels. *Advances in neural information processing systems*, 15, 2002. 9

[25] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020. 9

[26] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2426–2435, 2022. 9

[27] Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Diffusion models already have a semantic latent space. *arXiv preprint arXiv:2210.10960*, 2022. 9

[28] Changchun Li, Ximing Li, Jihong Ouyang, and Yiming Wang. Detecting the fake candidate instances: Ambiguous label learning with generative adversarial networks. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 903–912, 2021. 9

[29] Pengzhi Li, QInxuan Huang, Yikang Ding, and Zhiheng Li. Layerdiffusion: Layered controlled image editing with diffusion models. *arXiv e-prints*, pages arXiv–2305, 2023. 9

[30] Shikun Li, Xiaobo Xia, Shiming Ge, and Tongliang Liu. Selective-supervised contrastive learning with noisy labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 316–325, 2022. 9

[31] Sijia Li, Chen Chen, and Haonan Lu. Moecontroller: Instruction-based arbitrary image manipulation with mixture-of-expert controllers. *arXiv preprint arXiv:2309.04372*, 2023. 9

[32] Liping Liu and Thomas Dietterich. A conditional multinomial mixture model for superset label learning. *Advances in neural information processing systems*, 25, 2012. 9

[33] Liping Liu and Thomas Dietterich. Learnability of the superset label learning problem. In *International conference on machine learning*, pages 1629–1637. PMLR, 2014. 9

[34] 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, 3

[35] Gengyu Lyu, Yanan Wu, and Songhe Feng. Deep graph matching for partial label learning. In *IJCAI*, pages 3306–3312, 2022. 9

[36] Nam Nguyen and Rich Caruana. Classification with partial labels. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–559, 2008. 9

[37] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 9

[38] Zhihong Pan, Riccardo Gherardi, Xiufeng Xie, and Stephen Huang. Effective real image editing with accelerated iterative diffusion inversion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15912–15921, 2023. 9

[39] Or Patashnik, Daniel Garibi, Idan Azuri, Hadar Averbuch-Elor, and Daniel Cohen-Or. Localizing object-level shape variations with text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23051–23061, 2023. 9

[40] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10619–10629, 2022. 9

[41] Congyu Qiao, Ning Xu, and Xin Geng. Decompositional generation process for instance-dependent partial label learning. *arXiv preprint arXiv:2204.03845*, 2022. 2, 3, 9

[42] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020. 6

[43] Jianfei Ruan, Qinghua Zheng, Rui Zhao, and Bo Dong. Biased complementary-label learning without true labels. *IEEE Transactions on Neural Networks and Learning Systems*, 35 (2):2616–2627, 2022. 9

[44] 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. 9

[45] Cai-Zhi Tang and Min-Ling Zhang. Confidence-rated discriminative partial label learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. 9

[46] Shiyu Tian, Hongxin Wei, Yiqun Wang, and Lei Feng. Crosel: Cross selection of confident pseudo labels for partial-label learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19479–19488, 2024. 9

[47] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2023. 9

[48] Bram Wallace, Akash Gokul, and Nikhil Naik. Edict: Exact diffusion inversion via coupled transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22532–22541, 2023. 9

[49] Deng-Bao Wang, Li Li, and Min-Ling Zhang. Adaptive graph guided disambiguation for partial label learning. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 83–91, 2019. 9

[50] 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, 3, 9

[51] Zhizhong Wang, Lei Zhao, and Wei Xing. Stylediffusion: Controllable disentangled style transfer via diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7677–7689, 2023. 9

[52] 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. 1, 2, 3, 9

[53] 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, 3

[54] Dong-Dong Wu, Deng-Bao Wang, and Min-Ling Zhang. Distilling reliable knowledge for instance-dependent partial label learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 15888–15896, 2024. 2, 3, 9

[55] Qiucheng Wu, Yujian Liu, Handong Zhao, Ajinkya Kale, Trung Bui, Tong Yu, Zhe Lin, Yang Zhang, and Shiyu Chang. Uncovering the disentanglement capability in text-to-image diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1900–1910, 2023. 9

[56] Shiyu Xia, Jiaqi Lv, Ning Xu, and Xin Geng. Ambiguity-induced contrastive learning for instance-dependent partial label learning. In *IJCAI*, pages 3615–3621, 2022. 2, 3, 9

[57] Ming-Kun Xie, Feng Sun, and Sheng-Jun Huang. Partial multi-label learning with meta disambiguation. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 1904–1912, 2021. 9

[58] Ning Xu, Congyu Qiao, Xin Geng, and Min-Ling Zhang. Instance-dependent partial label learning. *Advances in Neural Information Processing Systems*, 34:27119–27130, 2021. 2, 3, 9

[59] Ning Xu, Biao Liu, Jiaqi Lv, Congyu Qiao, and Xin Geng. Progressive purification for instance-dependent partial label learning. In *International Conference on Machine Learning*, pages 38551–38565. PMLR, 2023. 2, 3, 9

[60] Sihan Xu, Ziqiao Ma, Yidong Huang, Honglak Lee, and Joyce Chai. Cyclenet: Rethinking cycle consistency in text-guided diffusion for image manipulation. *Advances in Neural Information Processing Systems*, 36, 2024. 9

[61] Fuchao Yang, Jianhong Cheng, Hui Liu, Yongqiang Dong, Yuheng Jia, and Junhui Hou. Mixed blessing: Class-wise embedding guided instance-dependent partial label learning. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 1763–1772, 2025. 3

[62] Zebin You, Yong Zhong, Fan Bao, Jiacheng Sun, Chongxuan Li, and Jun Zhu. Diffusion models and semi-supervised learners benefit mutually with few labels. *Advances in Neural Information Processing Systems*, 36, 2024. 9

[63] Fei Yu and Min-Ling Zhang. Maximum margin partial label learning. In *Asian conference on machine learning*, pages 96–111. PMLR, 2016. 9

[64] Fei Zhang, Lei Feng, Bo Han, Tongliang Liu, Gang Niu, Tao Qin, and Masashi Sugiyama. Exploiting class activation value for partial-label learning. In *International conference on learning representations*, 2021. 2, 3

[65] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36, 2024. 9

[66] Min-Ling Zhang and Fei Yu. Solving the partial label learning problem: An instance-based approach. In *IJCAI*, pages 4048–4054, 2015. 9

[67] Shu Zhang, Xinyi Yang, Yihao Feng, Can Qin, Chia-Chih Chen, Ning Yu, Zeyuan Chen, Huan Wang, Silvio Savarese, Stefano Ermon, et al. Hive: Harnessing human feedback for instructional visual editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9026–9036, 2024. 9

[68] Min Zhao, Fan Bao, Chongxuan Li, and Jun Zhu. Egsde: Unpaired image-to-image translation via energy-guided stochastic differential equations. *Advances in Neural Information Processing Systems*, 35:3609–3623, 2022. 9