Margin-Aware Prototype Debiasing for Generalized Category Discovery

Xinzi Cao, Feidiao Yang, Xiawu Zheng, Quanmin Liang, Yutong Lu, Yonghong Tian, *Fellow, IEEE*

***Abstract*— Generalized Category Discovery (GCD) is a challenging task that aims to identify both seen and novel categories in unlabeled data. We argue that a clear margin between seen and novel class representations is essential for accurate recognition. However, existing methods often ignore this margin, mapping representations to prototypes without enforcing separation between seen and novel classes. This leads to a bias where seen samples are misclassified as novel. To address this issue, we propose DebiasGCD, a debiasing framework that enhances prototype separation through margin-aware learning. Unlike prior work that relies on static prototype learning and overlooks fine-grained representations, our method introduces Dynamic Prototype Debiasing (DPD) and Spatial-Aware Representation Distillation (SARD) to mitigate this bias. First, DPD dynamically enforces inter-prototype margins, improving class-specific feature learning and prototype discrimination. Meanwhile, SARD promotes local representation of spatial learning, supporting DPD to capture subtle details that further refine class-specific features. By synergizing these components, DebiasGCD significantly improves prototype discriminability, generating more reliable predictions for seen classes. Extensive experiments demonstrate that our approach effectively mitigates pseudo-labeling bias across datasets, especially on fine-grained ones, achieving +8.3% and +9.6% improvements on the 'All' classes in CUB and Stanford Cars, respectively.**

***Index Terms*—Generalized Category Discovery, Novel Class Discovery, Open-set Recognition, Prototype Learning.**

I. Introduction

High-quality annotated data have been crucially advancing the development of artificial intelligence, particularly deep learning [1], [2], [3], [4], [5], [6]. However, in the era of big data and large models, massive amounts of data are in demand and certainly the vast majority of them are unlabeled. Since the acquisition of extensive annotations is costly, exploring the structure of unlabeled data with the help of some labeled data is an important and valuable research problem. To this end, semi-supervised learning (SSL) [7], [8], [9], [10] has emerged as a promising approach, which integrates both labeled and unlabeled data to enhance learning. However, SSL is based on a closed-world assumption that all samples must belong to known categories, which does not hold in some real-world scenarios where novel categories inevitably appear. To overcome this issue, *Generalized Category Discovery* (GCD) extends SSL by explicitly handling novel categories in unlabeled data. Therefore, GCD simultaneously classifies known categories and discovers novel ones, making it far more practical for real-world applications.

The core objective of GCD is to preserve a clear margin between the representations of seen and novel categories, thereby enabling reliable recognition. To achieve this, prior works [12], [13], [14], [15], [16], [17] approach the GCD problem from two main perspectives: (1) learning generic feature representations to enhance novel category discovery, and (2) generating pseudo-clusters or pseudo-labels for unlabeled data to guide classifier learning. The first approach typically involves contrastive learning [4], which improves the generalization ability of representations, while the second employs prototype-based classifiers built upon extracted features and jointly trained with labeled data and pseudo-labels derived from the representations. This paradigm is effective in leveraging discriminative representations and has been recognized as a key milestone in GCD.

Despite recent advances, our detailed investigation reveals a critical bias: *some seen-category samples are incorrectly labeled as novel, which results in imbalanced pseudo-labels for prototype training.* This issue is illustrated in Fig. 1a, where the left panel shows the t-SNE visualization of sample representations and the right panel depicts the class-center distance heatmap (darker colors denote smaller distances). Since the prototypes are narrowly separated, instances from seen categories may fall closer to novel-class prototypes and are thus misclassified (e.g., class 70 $\rightarrow$ class 25). Consequently, this bias not only persists but also intensifies as training progresses in the absence of external intervention. As shown in Fig. 1c, the pseudo-label accuracy of SimGCD stagnates at around 65% (blue curve), clearly indicating that the model fails to self-correct, while the bias continually reinforces itself.

Since pseudo-labels are generated by calculating image features and class prototypes, we identify two key sources of this bias: *Static Prototype Learning* and *Naive Representation Distillation*. First, while training the prototype classifier exclusively on labeled data from seen categories can achieve good performance on seen classes, this approach fails to distinguish seen from novel classes. Consequently, the learned prototypes lack discriminative power, which leads to biased pseudo-

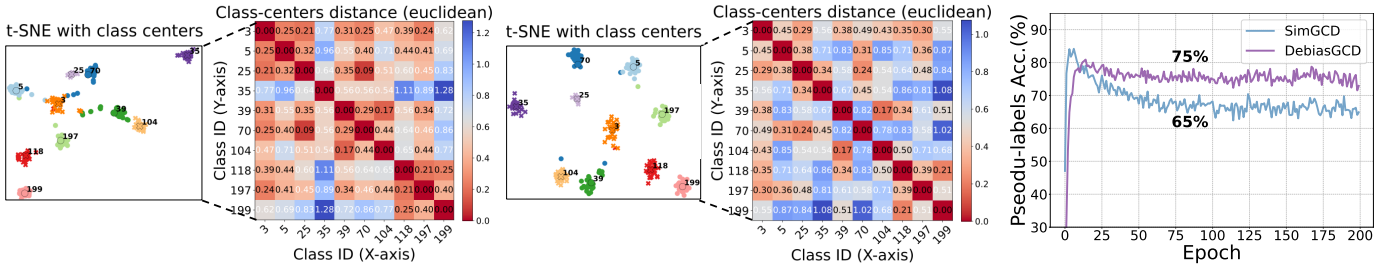
Xinzi Cao, Quanmin Liang, and Yutong Lu are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China (e-mail: caoxz@mail2.sysu.edu.cn; liangqm5@mail2.sysu.edu.cn; luyutong@mail.sysu.edu.cn).

Xiawu Zheng is with the Key Laboratory of Multimedia Trusted Perception and Efficient Computing, Ministry of Education of China, School of Informatics, Xiamen University, Xiamen 361005, China (e-mail: zhengxiawu@xmu.edu.cn).

Yonghong Tian is with the National Engineering Laboratory for Video Technology (NELVT), School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: yhtian@pku.edu.cn).

Xinzi Cao, Feidiao Yang, Quanmin Liang, and Yonghong Tian are also with Peng Cheng Laboratory, Shenzhen 518066, China (e-mail: yangfd@pcl.ac.cn).

Corresponding author: Yutong Lu.



(a) SimGCD with bias (b) Ours with less bias (c) Pseudo-labels accuracy in CUB.

Fig. 1: Overview of the pseudo-labels bias in CUB [11] caused by unclear margins. (a)–(b) visualize class-center distances from t-SNE, where darker colors indicate shorter distances (diagonal = self-distance). In (a), small inter-class distances (red) cause misclassifications of a seen sample (e.g., class 70 → class 25). In (b), our margin-aware debiasing enlarges inter-class margins and correctly identifying the sample as seen. (c) shows SimGCD suffers from low pseudo-label accuracy, while our method improves it by 10%. Notably, pseudo-label accuracy measures the teacher's predictions on seen samples within the unlabeled data for self-distillation.

labeling toward novel categories. Second, existing prototype classifiers [15], [18], [17], [16] rely solely on global representations (i.e., class token) for prototype learning [4], [19], while completely overlooking local features (i.e., patch token) that are critical for fine-grained classification. This limitation restricts the model's ability to learn discriminative representations, thereby further degrading pseudo-label quality.

To address this bias, we propose DebiasGCD, a debiasing framework that calibrates bias through margin-aware prototype learning. Unlike previous methods that rely on static prototype mapping and ignore the boundaries between prototypes, we introduce Dynamic Prototype Debiasing (DPD), which enforces a clear margin between prototypes. This encourages the model to learn more discriminative representations, strengthens class-specific features, and ultimately improves the reliability of pseudo-labels. Additionally, instead of relying solely on global representations, we propose the Spatial-Aware Representation Distillation (SARD) module, which transfers fine-grained knowledge by transforming patch tokens into spatial feature maps, thereby enhancing representation learning and improving DPD optimization. Furthermore, we apply both strong and weak augmentations, as proposed in [18], [8], which help generate more stable pseudo-labels and contribute to more stable debiasing. Finally, by combining DPD and SARD, DebiasGCD establishes large margins between seen and novel classes, effectively mitigating pseudo-labeling bias.

To evaluate the effectiveness of our debiasing framework, we conduct extensive experiments on seven datasets, including fine-grained and generic object classification datasets. Overall, our approach significantly reduces pseudo-labeling bias, prevents a **10**% drop in pseudo-label accuracy (Fig. 1c), and produces a clearer margin compared to the baseline [15] SimGCD (Fig. 1b *vs.* Fig. 1a). By leveraging these reliable pseudo-labels, our debiasing framework achieves competitive performance in the GCD task across all datasets.

In summary, our key contributions are as follows:

- We investigate and identify a previously unaware bias in prototype classifiers, which results in imbalanced pseudo-labeling in Generalized Category Discovery.
- We introduce DebiasGCD, a margin-aware prototype debiasing approach that utilizes DPD and SARD to maintain inter-prototype margin and extract more discriminative local representation for identifying instances within unlabeled data.
- Through extensive experiments, we demonstrate that our debiasing method effectively mitigates bias, achieving significant performance improvements—*e.g.*, **+8.3%** and **+9.6%** gains on 'All' classes in the CUB and Stanford Cars, respectively.

II. Related Work

A. Semi-supervised Learning

SSL tackles the issue of limited labeled data by integrating unlabeled data from pre-defined classes [20], [21], [22]. Most SSL methods adopt techniques like consistency-based regularization [23], [24], [25], [26], pseudo-labeling [8], [27], [28], and transfer learning [29], [30]. Notably, pseudo-labeling is an effective baseline using a weak augmented view's prediction as a pseudo-label for a strong view [31], [8]. Meanwhile, [32], [33], [34] improves performance by adjusting thresholds to select high-quality pseudo-labels. However, traditional SSL methods assume the labeled and unlabeled data share the same distribution, limiting its effectiveness in open-world settings. Recently, SSL has been extended to out-of-distribution (OOD) detection for generating high-quality pseudo-labels [35], [28], [36], [37]. For example, InPL [28] employs energy-based pseudo-labeling for OOD data, HIMPLoS [38] reduces overconfidence by using cosine similarity with class prototypes as an adaptive logit temperature, and PSA [39] adopts a dual-threshold ternary assignment strategy to filter uncertain samples, thereby ensuring higher-purity pseudo-labels. However, these methods focus on improving pseudo-label quality by filtering out OOD samples, which differs from the goal of GCD. Unlike OOD detection, GCD requires leveraging both ID and OOD data: we treat OOD samples as unlabeled unseen categories and explicitly mitigate the imbalance caused when seen samples (ID) are incorrectly classified as unseen (OOD).

B. Generalized Category Discovery

The goal of Generalized Category Discovery (GCD) [12] is to train a model to recognize both seen and novel categories within unlabeled data. Traditional GCD [12] recognizes the unlabeled data using *k*-means clustering their representation. GPC [13] uses a Gaussian Mixture Model (GMM) and representation learning to cluster categories. PromptCAL [40] enhances semantic representations with prompt learning and contrastive affinity, but still relies on SemiKMeans clustering [12]. XCon [41] propose part the dataset via k-means and performs contrastive learning within each subset to capture fine-grained discriminative features. Despite their effectiveness, these methods are computationally expensive for large datasets. To address this, parametric classifier methods have emerged. SimGCD [15] introduces a prototype classifier establishing a new GCD baseline. μGCD [18] utilizes mean-teacher and strong/weak augmentations to enhance pseudo-label quality. LegoGCD [17] tackles catastrophic forgetting in SimGCD via entropy regularization. SPNet [16] further optimizes SimGCD with data prompting. PartGCD [42] enhances semantic separation by extracting part features and incorporating them into learning, but introduces additional computational overhead. More recently, GET [43] leverages CLIP [44] to introduce multi-modal information into GCD, achieving strong performance. However, this approach depends on large-scale pre-trained models, adding significant computational and memory costs. In this work, we aim to build on the strong baseline SimGCD and improve pseudo-label quality for prototype classifiers, addressing a key limitation of existing methods.

C. Prototype Learning

Prototype learning treats class-specific representations in the feature space as prototype centers for each category [45], matching instances to the category with the highest similarity to its prototype. For instance, [46] separates seen and novel categories using a decoupled prototypical network, aligning them in labeled and unlabeled data to transfer knowledge and capture semantics. TAN [47] trains a model to align instances with prototypes and estimate novel prototypes in unlabeled data based on category similarities. Prototype learning has also been explored for separating seen and unseen domains. ICPC [48] adopts a two-stage pipeline: first calibrating class prototypes to approximate the true feature centers, and then retraining the model on relatively reliable unseen samples aligned with predefined prototypes. However, this incurs additional training overhead. Similarly, DPN [49] introduces a dual-attention prototype framework to capture fine-grained visual cues, but the dual-network design leads to high computational cost. In contrast, we adopt a parametric prototype-based classifier following SimGCD [15], and propose SARD to efficiently learn class-specific fine-grained features without relying on costly dual-stage or dual-network designs.

III. PRELIMINARIES

A. Problem Formulation

Suppose $\mathcal{X}$ is the input space, we assume the labeled dataset $\mathcal{D}^l = \{(\boldsymbol{x}, y)\} \in \mathcal{X} \times \mathcal{Y}^l$, containing only known categories, and the unlabeled dataset $\mathcal{D}^u = \{(\boldsymbol{x}, y)\} \in \mathcal{X} \times \mathcal{Y}^u$, which includes both seen and novel categories, where $\mathcal{Y}^l \subset \mathcal{Y}^u$. The objective of GCD is to categorize the samples in unlabeled data $\mathcal{D}^u$, using the labels from known categories ($\mathcal{Y}^l$) and unlabeled data ($\mathcal{D}^u$). Notably, the total number of known and novel categories is denoted as $K = |\mathcal{Y}^u|$, and K is known from prior works [15], [50], [51].

B. Architecture

Following SimGCD [15], we construct a model f_θ with parameter θ to map an input image to a label in $\mathcal{Y}_u$. The model consists of an image encoder Φ, a projection head h, and a classification head g. For a sample $\boldsymbol{x}_i$, its feature representation is $\boldsymbol{z}_i = \Phi(\boldsymbol{x}_i) \in \mathbb{R}^{(n+1)\times d}$, where n is the number of patches, 1 corresponds to the class token, and d is the feature space dimension. The number of patches n is computed as $n = \frac{H\times W}{h\times w}$, where H and W are image dimensions, and h and w are the patch sizes.

C. Representation Learning.

Following GCD [12] and SimGCD [15], we adopt contrastive learning [4] to extract feature representations for categorizing unlabeled data. Specifically, we perform supervised contrastive learning [52] on labeled data $\mathcal{D}^l$ and self-supervised contrastive learning [53] on the entire dataset. The overall representation loss is denoted as $\mathcal{L}_{rep}(\theta; \mathcal{D})$.

Formally, given two views x_i and x_i' of the same image in a all data $\mathcal{D}$, the parameters of feature extractor Φ can be updated by the InfoNCE loss [54] in self-supervised contrastive learning:

$$\mathcal{L}_{rep}(\theta; \mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x}_i \in \mathcal{D}} \log \frac{\exp\left\langle \boldsymbol{z}_i^{cls}, \boldsymbol{z}_i^{cls\prime} \right\rangle / \tau}{\sum_n^{n\neq i} \exp\left\langle \boldsymbol{z}_i^{cls}, \boldsymbol{z}_n^{cls} \right\rangle / \tau}, \quad (1)$$

where $\boldsymbol{z}^{cls}$ is the first row vector in its feature representation $\boldsymbol{z}_i = h \circ (\Phi(x_i)$, denoted as $\boldsymbol{z}_i^{cls}$. τ is a temperature hyperparameter. Analogous to Eq. (1), we use supervised contrastive loss in the same class as:

$$\mathcal{L}_{rep}(\theta; \mathcal{D}^l) = -\frac{1}{|\mathcal{D}^l|} \sum_{\boldsymbol{x}_i \in \mathcal{D}} -\frac{1}{|\mathcal{N}(i)|} \sum_{q \in \mathcal{N}(i)} \log \frac{\exp\left\langle \boldsymbol{z}_i^{cls}, \boldsymbol{z}_q^{cls} \right\rangle / \tau}{\sum_n^{n\neq i} \exp\left\langle \boldsymbol{z}_i^{cls}, \boldsymbol{z}_n^{cls} \right\rangle / \tau}. \quad (2)$$

Finally, the total contrastive loss on the model's representation is given as:

$$\mathcal{L}_{rep}(\theta; \mathcal{D}) = (1 - \lambda_1)\, \mathcal{L}_{rep}(\theta; \mathcal{D}^u) + \lambda_1 \mathcal{L}_{\text{rep}}(\theta; \mathcal{D}^l). \quad (3)$$

D. Prototype Classifier

Unlike GCD [12], which uses a time-intensive clustering-based approach, SimGCD [15] designs an effective prototypical classifier g with parameter $\mathbf{W} \in \mathbb{R}^{d\times K}$ based on self-distillation [4], [19]. The column vectors in $\mathbf{W} = [w_1, \ldots, w_K]$ represents K prototypes, each corresponding to a category. Given a sample view $\boldsymbol{x}$, its feature representation $\boldsymbol{z}$ is extracted, where the first row, known as the class token,

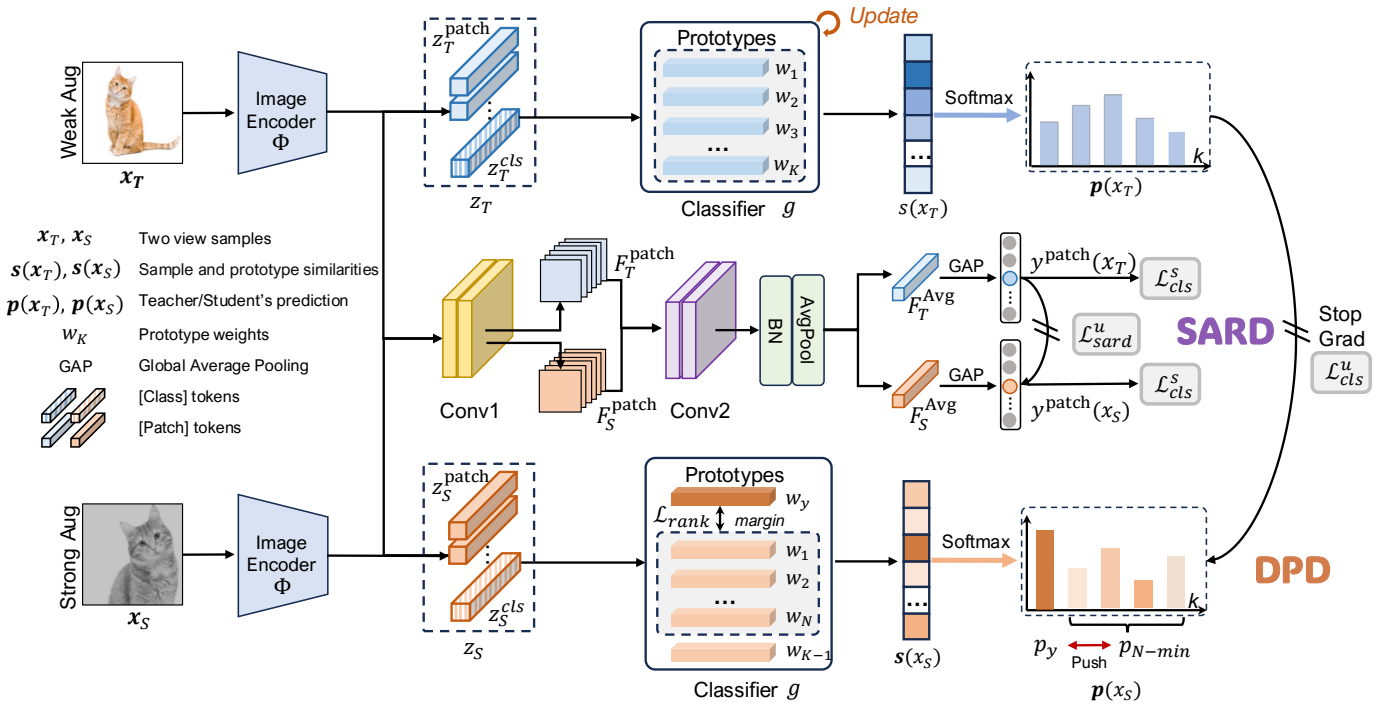


Fig. 2: The overall framework of DebiasGCD consists of two key components. First, the Dynamic Prototype Debiasing (DPD) enhances prototype discriminability by leveraging the ranking loss $\mathcal{L}_{\text{rank}}$, which enforces a margin between the ground-truth prototype $\boldsymbol{w}_y$ and the N-th smallest probability prototype $\boldsymbol{w}_N$. Next, Spatial-Aware Representation Distillation (SARD) converts local patch tokens into spatial feature maps, refines them using convolutional layers, and aggregates spatially-aware predictions, enhancing local feature learning in DPD to better capture category-specific features. Finally, the teacher network prototypes are updated online from the student during training.

is denoted by $\boldsymbol{z}^{\text{cls}}$. The cosine similarities $\boldsymbol{s}(\boldsymbol{x})$ between the class token and the prototypes are computed as:

$$\begin{aligned}\boldsymbol{s}(\boldsymbol{x}) &= (s_1, s_2, \ldots, s_K) \\ &= \left(\langle \mathbf{w}_1, \boldsymbol{z}^{cls}\rangle, \langle \mathbf{w}_2, \boldsymbol{z}^{cls}\rangle, \ldots, \langle \mathbf{w}_K, \boldsymbol{z}^{cls}\rangle\right).\end{aligned} \tag{4}$$

Then, the softmax function is applied to the similarities to imitate the traditional classifier to obtain pseudo-logits:

$$\boldsymbol{p}(\boldsymbol{x}) = \frac{\exp\left(s(\boldsymbol{x})/\tau\right)}{\sum_{j=1}^{K} \exp\left(s(\boldsymbol{x})_j/\tau\right)}. \tag{5}$$

Next, the prototype classifier is trained using a cross-entropy loss with pseudo-labels in an **un**supervised manner:

$$\mathcal{L}_{\text{cls}}^{u}(\theta;\mathcal{D}) = -\frac{1}{|\mathcal{D}|}\sum_{\boldsymbol{x}\in\mathcal{D}} \boldsymbol{p}(\boldsymbol{x}_T)\log \boldsymbol{p}(\boldsymbol{x}_S), \tag{6}$$

where $\boldsymbol{p}(\boldsymbol{x}_T)$ and $\boldsymbol{p}(\boldsymbol{x}_S)$ are the logits of views $\boldsymbol{x}_T$ and $\boldsymbol{x}_S$ from image $\boldsymbol{x}$, forwarded through the teacher and student networks, respectively. Meanwhile, SimGCD also adopts a standard cross-entropy loss $\mathcal{L}_{\text{cls}}^{s}$ on labeled data and a class mean entropy regulariser [19], defined as $H(\overline{\boldsymbol{p}}) = -\sum_k \overline{\boldsymbol{p}}^{(k)} \log \overline{\boldsymbol{p}}^{(k)}$, where $\overline{\boldsymbol{p}}$ is the mean prediction of each class in a batch, computed as $\overline{\boldsymbol{p}} = \frac{1}{2|B|}\sum_{i\in B}(\boldsymbol{p}(\boldsymbol{x}_S) + \boldsymbol{p}(\boldsymbol{x}_T))$. Therefore, the overall training classification loss is:

$$\mathcal{L}_{\text{cls}}(\theta;\mathcal{D}) = (1-\lambda)(\mathcal{L}_{\text{cls}}^{u}(\theta;\mathcal{D}) - \varepsilon H(\overline{\boldsymbol{p}})) + \lambda\mathcal{L}_{\text{cls}}^{s}(\theta;\mathcal{D}^{l}). \tag{7}$$

Finally, the overall objective in baseline is $\mathcal{L}_{\text{rep}}(\theta;\mathcal{D}) + \mathcal{L}_{\text{cls}}(\theta;\mathcal{D})$.

E. Bias in Prototype Classifier

While prototype-based classifiers [15] have demonstrated strong performance, they suffer from a critical bias: during semi-supervised learning, some samples from seen categories in the unlabeled data are misclassified as novel categories (see Fig. 1c). This bias arises from insufficient similarity between the sample's class token and the class prototypes. Specifically, the pseudo-label $\boldsymbol{p}(\boldsymbol{x})$ in Eq. (5) is assigned based on this similarity; however, when prototypes are not well-separated, the model often produces incorrect pseudo-labels. These noisy labels are then propagated to the student model via Eq. (6), further reinforcing the bias and leading to compounding misclassifications. Over time, this creates a self-reinforcing loop of error, as evidenced by the persistent degradation shown by the blue curve in Fig. 1c.

IV. Method

In this section, we introduce our DebiasGCD approach for mitigating bias in pseudo-labeling. The overall framework of our method is depicted in Fig. 1 and comprises two components: a Dynamic Prototype Debiasing (DPD) module (Sec.IV-A) and a Spatial-Aware Representation Distillation (SARD) module (Sec. IV-B).

A. Dynamic Prototype Debiasing

1) Motivation: A strategy to mitigate this bias is to enforce clear margins between prototypes. However, directly modify-

ing the distance between prototypes is intractable. Instead, we work around the issue by adjusting the pseudo-labels $\boldsymbol{p}(\boldsymbol{x})$ generated by the prototypes, which enables gradient flow through the similarity computation and allows back-propagation to gradually reduce the bias during training. Inspired by [55], which incorporates adaptive margins into model predictions $\boldsymbol{p}(\boldsymbol{x})$ to reduce inter-class confusion, we propose a more fine-grained approach by ranking predictions using margins. To achieve this, we introduce a ranking-based loss $\mathcal{L}_{\text{rank}}$, which enforces a margin between the prediction of the ground-truth label and the remaining predictions on labeled data. As a result, this indirectly encourages ground-truth prototypes to remain well-separated from others, thereby enhancing their discriminative ability.

2) Mechanism: To implement this ranking mechanism, the ranking loss $\mathcal{L}_{\text{rank}}$ involves three steps: (1) First, extract the ground-truth prediction from the student output $\boldsymbol{p}(\boldsymbol{x}_S)$. (2) Next, sort the remaining probabilities in ascending order and select the N-th smallest probability $p_{N\text{-min}}$. (3) Finally, compare the ground-truth prototype w_y with the competing prototype w_N (corresponding to $p_{N\text{-min}}$) using margin ranking loss $\mathcal{L}_{\text{rank}}$. The detailed formulation is presented below.

Step (1): Extracting the ground-truth prediction p_y from $\boldsymbol{p}(\boldsymbol{x}_S)$.

Given the prediction vector output $\boldsymbol{p}\left(\boldsymbol{x}_S\right) = [p_1, p_2, \ldots, p_K]$ by the student network from Eq. (5), we first extract the probability corresponding to the ground-truth class label $y \in \mathcal{Y}^l$: $p_y = \boldsymbol{p}\left(\boldsymbol{x}_S\right)[y]$.

Step (2): Sorting the remaining probabilities and selecting the N-th smallest.

To identify the most challenging competing prototype, we first remove the ground-truth probability from the prediction vector, and sort this set in ascending order:

$$\boldsymbol{p}^{DPD}(\boldsymbol{x}_S) = \text{sort}\Big(\{p_j \mid j \neq y,\, j \in [1, K-1]\}\Big), \quad (8)$$

After sorting, we select the N-the smallest probability:

$$p_{N-\min} = p_{\pi(N)}, \quad (9)$$

where $\pi(N)$ represents the index of the N-th smallest probability. Here, the hyperparameter N is chosen such that $N \leq K-1$.

Step (3): Increasing the margin between the ground-truth prototype and competing prototypes.

Directly optimizing the prototypes is not feasible, therefore, we compare them indirectly by evaluating their corresponding predictions. Specifically, w_y corresponds to p_y, while the prototype w_N corresponds to the selected probability $p_{N\text{-min}}$. Based on this comparison, the loss function is formulated as:

$$\mathcal{L}_{\text{rank}}\ \left(\theta; \mathcal{D}^l\right) = \frac{1}{|\mathcal{D}^l|} \sum_{\boldsymbol{x} \in \mathcal{D}^l} \max\left(0, -p_y + p_{N\text{-min}} + \epsilon\right), \quad (10)$$

where ϵ is a predefined margin threshold. This enforces $p_y \geq p_{N\text{-min}} + \epsilon$. If p_y is smaller than $p_{N\text{-min}}$, or does not exceed it by at least the margin ϵ, then $p_{N\text{-min}} - p_y + \epsilon$ is positive. In this case, the loss will be non-zero, and the model parameters are updated to increase p_y relative to $p_{N\text{-min}}$. Otherwise, if p_y is sufficiently larger than $p_{N\text{-min}}$, the term $p_{N\text{-min}} - p_y + \epsilon$ becomes negative, resulting in a zero loss, and the model receives no gradient update. By minimizing $\mathcal{L}_{\text{rank}}(\theta; \mathcal{D}^l)$, a margin is enforced between the correct prototype w_y and the low-ranked prototype w_N (corresponding to the N-th smallest probability), thereby enhancing prototype discriminability and improving pseudo-labeling accuracy.

3) Empirical Analysis: We empirically investigate why the smallest remaining probability ($p_{N\text{-min}}$) is preferred over the highest one ($p_{N\text{-max}}$) in Eq. (10). As illustrated in Fig. 8, the variants employing $p_{N\text{-min}}$ (blue bars) consistently outperform those using $p_{N\text{-max}}$ (purple bars) across all datasets and evaluation splits (*All*, *Old*, and *New*). This performance gap remains stable for $N \in \{1, 3, 5\}$, suggesting that emphasizing the easiest negative class—i.e., the one with the smallest non-target probability—provides more reliable and noise-resistant optimization signals. In contrast, focusing on $p_{N\text{-max}}$ tends to overemphasize hard negatives, which, while slightly surpassing the SimGCD baseline, still fall behind the variants using $p_{N\text{-min}}$. Consequently, we adopt $p_{N\text{-min}}$ in Eq. (10) to construct a more effective margin ranking loss that enhances model robustness and optimization stability.

B. Spatial-Aware Representation Distillation

Unlike prior prototype-based methods [15], [16] that rely only on global representations, our approach explicitly incorporates local features. As shown in Eq. (4) and Eq. (5), pseudo-labels are associated with both prototypes and the image's class token. However, the class token mainly captures global image-level information and misses fine-grained details. This is especially problematic in datasets like CUB [11], where visually similar classes can only be distinguished by subtle local cues (e.g., bill shape). To address this, we emphasize learning from local representations, which directly refine pseudo-labels and also strengthen prototype learning, thereby improving classifier discriminability for DPD.

To this end, we propose a lightweight Spatial-Aware Representation Distillation (SARD) module to preserve semantic consistency in local features. As shown in Fig. 2, SARD reshapes patch tokens into a spatial feature map, applies convolutional layers (Conv1 and Conv2) to capture local patterns, and aggregates predictions via global average pooling. By combining transformer-based global modeling with local feature extraction, SARD enhances representation learning with improved spatial awareness.

Given the patch tokens $\boldsymbol{z}^{\text{patch}} \in \mathbb{R}^{n \times d}$ extracted from the image encoder Φ, where n denotes the number of patches (see Sec. III-B), we aim to capture local spatial features. To this end, we first pass $\boldsymbol{z}^{\text{patch}}$ through a convolutional layer (Conv1) to extract fine-grained local information:

$$\boldsymbol{F}^{\text{patch}} = \text{Conv}\left(\boldsymbol{z}^{\text{patch}}; \mathbf{W}_{C1}\right), \quad (11)$$

where $\mathbf{W}_{C1}$ represents the learnable parameters of Conv1. This step uses a small 3×3 kernel to extract local spatial features, thereby enhancing spatial awareness at a fine-grained level. Next, the extracted feature $\boldsymbol{F}^{\text{patch}}$ is further processed

Algorithm 1 One training Procedure for DebiasGCD.

Input: Encoder Φ, classifier head g, projector h, labeled data $\mathcal{D}^l$, unlabeled data $\mathcal{D}^u$, N-th.
Output: Φ, g, and h.
for $(\boldsymbol{x}_S, \boldsymbol{x}_T) \in \mathcal{D}^l \cup \mathcal{D}^u$, $y \in \mathcal{Y}_l$ **do**
$\boldsymbol{p}(\boldsymbol{x}_S) \leftarrow f \circ \Phi(\boldsymbol{x}_S)$ using Eq. (5);
Step 1: Extract p_y from $\boldsymbol{p}(\boldsymbol{x}_S)$: $p_y = \boldsymbol{p}(\boldsymbol{x}_S)[y]$;
Step 2: Select the N-th pro.: $p_{N-\min}$ using Eq. (8)(9);
Step 3: Increase the distance between w_y and w_N via Eq. (10);
Capture local spatial feature $\boldsymbol{F}^{\text{Avg}}$ using Eq. (11)(12);
Compute the ranking-based loss $\mathcal{L}_{\text{rank}}(\theta; \mathcal{D}^l)$
Compute the SARD loss $\mathcal{L}^u_{\text{sard}}(\theta; \mathcal{D})$ using Eq. (13);
Add total loss $\mathcal{L}_{\text{total}} = \alpha \cdot \mathcal{L}^u_{\text{sard}}(\theta; \mathcal{D}) + \beta \cdot \mathcal{L}_{\text{rank}}(\theta; \mathcal{D}^l)$;
Update Φ, g, and h by SGD.
end for

by Conv2, which consists of a convolutional layer followed by batch normalization (BN) and average pooling (AvgPool). This sequence of operations generates the locally smoothed feature, formulated as:

$$\boldsymbol{F}^{\text{Avg}} = \text{AvgPool}_{2D}\left(\text{BN}\left(\text{Conv}\left(\boldsymbol{F}^{\text{patch}}; \mathbf{W}_{C2}\right)\right)\right), \quad (12)$$

where $\mathbf{W}_{C2}$ represents the learnable parameters of Conv2. The average feature $\boldsymbol{F}^{\text{Avg}}$ is passed through global average pooling (GAP) [56], yielding $\boldsymbol{y}^{\text{patch}}(\boldsymbol{x}) = GAP(\boldsymbol{F}^{\text{Avg}})$, which aggregates spatial information into a compact global representation, preserving the full spatial distribution while generating the class probability distribution.

Similar to the classification in Eq. (6), we apply an entropy loss between weak and strong views to align the information of local features:

$$\mathcal{L}^u_{\text{sard}}(\theta; \mathcal{D}) = -\frac{1}{|\mathcal{D}|}\sum_{\boldsymbol{x}\in\mathcal{D}} \boldsymbol{y}^{\text{patch}}(\boldsymbol{x}_T) \log \boldsymbol{y}^{\text{patch}}(\boldsymbol{x}_S), \quad (13)$$

where the probability distribution $\boldsymbol{y}^{\text{patch}}(\boldsymbol{x}_S)$ is from another sample $\boldsymbol{x}_S$. This objective encourages the model to focus on detailed features of the object, helping it learn more fine-grained information.

C. Overall Learning Objectives

We propose DebiasGCD, a debiasing pseudo-labeling approach for GCD, by integrating the Dynamic Prototype Debiasing (DPD) and Spatial-Aware Representation Distillation (SARD) modules. The overall training loss for debiasing is formulated as:

$$\mathcal{L}_{\text{total}} = \underbrace{\mathcal{L}_{\text{rep}}(\theta; \mathcal{D}) + \mathcal{L}_{\text{cls}}(\theta; \mathcal{D})}_{\text{Baseline}} + \alpha \cdot \underbrace{\mathcal{L}^u_{\text{sard}}(\theta; \mathcal{D}) + \beta \cdot \mathcal{L}_{\text{rank}}(\theta; \mathcal{D}^l)}_{\text{Updated}}, \quad (14)$$

where α and β are balancing factors that control the learning of the prototypical classifier.

V. Experiments

A. Experimental Setup

1) Datasets: We evaluate our approach on seven datasets, including two generic object recognition benchmarks: CIFAR100 [57] and ImageNet-100 [1]; three semantic shift benchmarks (SSB) [12]: CUB [11], Stanford Cars [58], and FGVC-Aircraft [59]; as well as the imbalanced datasets Herbarium 19 [60], CIFAR-LT [61] and the challenging large-scale ImageNet-1K [1]. Following GCD [12], we randomly sample 50% of the seen categories from the training set to construct the labeled set $\mathcal{D}^l$, with the remaining images from seen and novel categories forming the unlabeled subset $\mathcal{D}^u$. Table I details our experimental split protocol.

TABLE I: Details of the benchmark datasets.

Dataset	Balance	Labeled $\mathcal{D}^l$		Unlabeled $\mathcal{D}^u$	
		Images	Class	Images	Class
CUB [11]	✓	1.5k	100	4.5k	200
Stanford Cars [58]	✓	2.0k	98	6.1k	196
FGVC-Aircraft [59]	✓	1.7k	50	5.0k	100
Herbarium 19 [60]	✗	8.9k	341	25.4k	683
CIFAR100 [57]	✓	20.0k	80	30.0k	100
ImageNet-100 [1]	✓	31.9k	50	95.3k	100
ImageNet-1K [1]	✓	321k	500	960k	1000

2) Evaluation protocols: Following GCD [12] evaluation protocol, we employ clustering accuracy (ACC) to evaluate the model performance.me Concretely, ACC is computed using the predicted labels $\hat{y}$ with ground truth labels y^*, defined as $ACC = \frac{1}{M}\sum_{i=1}^{M} \mathbb{I}(y_i^* = p(\hat{y}_i))$, where $M = |\mathcal{D}^u|$, and p aligns predicted cluster assignments with ground truth class labels using the Hungarian optimal assignment algorithm [62]. Notably, our results are reported on the unlabeled set $\mathcal{D}^u$, which contains samples from both known and novel classes. Here, 'All' denotes ACC on all samples in $\mathcal{D}^u$, 'Old' on known-class samples, and 'New' on novel-class samples.

3) Implementation details: In line with prior GCD works [12], [15], we use a ViT-B/16 backbone [3], pre-trained with DINO [4] and DINOv2 [63]. During training, we fine-tune only the last attention block across all datasets. We apply *strong/weak* data augmentation [8], using strong augmentation to the student network and weak augmentation to the teacher network, as described in [18]. Our setup includes a batch size of 128, 200 training epochs, and an initial learning rate of 0.1 with cosine decay. For classifier training, τ starts at 0.07 and is warmed up to 0.04 over the first 30 epochs. The ranking label $r = 1$ in Eq. (10) assumes that p_y ranks higher than $p_{N\text{-min}}$. For hyperparameters, N we consider the range $\{1,2,3,4,5\}$. Regardless of the specific value, our method consistently outperforms the baseline. In practice, the best results are obtained with $N \in \{3, 4\}$ and $\epsilon \in \{1, 2\}$. For fine-grained datasets, we set $\alpha = 1, \beta = 1$, while for coarse-grained datasets, $\alpha = 0.5, \beta = 0.5$, reflecting the need for stronger separation in fine-grained scenarios due to higher feature similarity. All the experimental results are obtained using the final student network. Experiments are implemented in PyTorch on Nvidia Tesla V100 GPUs.

B. Quantitative Comparison

1) Evaluation on fine-grained datasets: Identifying fine-grained samples in GCD is challenging because of their subtle inter-class differences, such as birds with similar heads but

TABLE II: Results on the Semantic Shift Benchmark [12].

	Method	CUB			Stanford Cars			FGVC-Aircraft		
		All	Old	New	All	Old	New	All	Old	New
DINO	*k*-means [64]	34.3	38.9	32.1	12.8	10.6	13.8	16.0	14.4	16.8
	RS+ [51]	33.3	51.6	24.2	28.3	61.8	12.1	26.9	36.4	22.2
	UNO+ [50]	35.1	49.0	28.1	35.5	70.5	18.6	40.3	56.4	32.2
	ORCA [65]	35.3	45.6	30.2	23.5	50.1	10.7	22.0	31.8	17.1
	GCD [12]	51.3	56.6	48.7	39.0	57.6	29.9	45.0	41.1	46.9
	DCCL [14]	63.5	60.8	64.9	-	-	-	43.1	55.7	36.2
	PromptCAL [40]	62.9	64.4	62.1	50.2	70.1	40.6	52.2	52.2	52.3
	SPTNet [16]	65.8	68.8	65.1	59.0	79.2	49.3	59.3	61.8	58.1
	LegoGCD [17]	63.8	71.9	59.8	57.3	75.7	48.4	55.0	61.5	51.7
	CMS [66]	68.2	76.5	64.0	56.0	63.4	52.3	56.9	**76.1**	47.6
	AptGCD [67]	70.3	74.3	69.2	62.1	79.7	53.6	**61.1**	65.2	59.0
	RLGCD [68]	70.0	**79.1**	65.4	64.9	79.3	58.0	60.6	62.2	**59.8**
	MOS [69]	69.6	72.3	68.2	64.6	80.9	56.7	**61.1**	66.9	58.2
	PALGCD [70]	**72.6**	75.2	**71.2**	**72.2**	**83.4**	**66.8**	58.8	64.5	56.0
	SimGCD [15]	60.3	65.6	57.7	53.8	71.9	45.0	54.2	59.1	51.8
	Hyp-SimGCD [71]	64.8	65.8	64.2	62.8	73.4	57.7	58.7	58.9	58.5
	SimGCD+AF [72]	69.0	74.3	66.3	67.0	80.7	**60.4**	59.4	68.1	55.0
	SimGCD+Debiasing (Ours)	68.6	72.6	66.6	63.4	81.0	54.9	58.4	66.0	54.2
DINOv2	CiPR [73]	78.3	73.4	**80.8**	66.7	77.0	61.8	-	-	-
	GCD [12]	71.9	71.2	72.3	65.7	67.8	64.7	55.4	47.9	59.2
	μGCD [18]	74.0	75.9	73.1	76.1	91.0	68.9	66.3	68.7	65.1
	RLGCD [68]	78.7	79.5	78.3	79.5	91.8	73.5	72.6	**77.3**	70.3
	SimGCD	71.5	78.1	68.3	71.5	81.9	66.6	63.9	69.9	60.9
	Hyp-SimGCD [71]	77.6	77.9	77.4	**82.5**	85.8	**81.0**	**76.4**	70.3	79.4
	SimGCD+Debiasing (Ours)	**79.2**	**81.3**	**76.1**	78.2	**91.8**	72.6	74.1	72.1	75.2

TABLE III: Results on generic image recognition datasets.

	Method	CIFAR100			ImageNet-100		
		All	Old	New	All	Old	New
DINO	*k*-means [64]	52.0	52.2	50.8	72.7	75.5	71.3
	RS+ [51]	58.2	77.6	19.3	37.1	61.6	24.8
	UNO+ [50]	69.5	80.6	47.2	70.3	95.0	57.9
	ORCA [65]	69.0	77.4	52.0	73.5	92.6	63.9
	GCD [12]	73.0	76.2	66.5	74.1	89.8	66.3
	DCCL [14]	75.3	76.8	70.2	80.5	90.5	76.2
	PromptCAL [40]	81.2	84.2	75.3	83.1	92.7	78.3
	SPTNet [16]	81.3	84.3	75.6	85.4	93.2	81.4
	LegoGCD [17]	81.8	81.4	82.5	86.3	94.5	82.1
	CMS [66]	82.3	**85.7**	75.5	84.7	**95.6**	79.2
	AptGCD [67]	82.8	81.8	81.8	**87.8**	95.4	**84.3**
	RLGCD [68]	83.4	84.2	81.9	86.9	94.2	83.2
	PALGCD [70]	78.5	78.9	77.8	83.0	93.1	78.0
	SimGCD [15]	80.1	81.2	77.8	83.0	93.1	77.9
	Hyp-SimGCD [71]	82.4	83.1	81.2	86.5	93.7	83.0
	SimGCD+AF [72]	82.2	85.0	76.5	85.4	94.6	80.8
	SimGCD+Debiasing (Ours)	**84.2**	84.3	**84.0**	85.0	94.6	80.2
DINOv2	CiPR [73]	90.3	89.0	93.1	88.2	87.6	88.5
	GCD [12]	79.6	84.5	69.9	78.5	89.5	73.0
	SPTNet [16]	-	-	-	90.1	96.1	87.1
	RLGCD [68]	91.2	91.2	91.2	**92.1**	96.2	90.0
	SimGCD [15]	88.5	89.2	87.2	89.9	95.5	87.1
	Hyp-SimGCD [71]	91.5	90.0	**94.6**	91.9	96.2	**89.8**
	SimGCD+Debiasing (Ours)	**93.5**	**94.0**	93.2	91.2	**96.6**	88.7

different bill sizes. As shown in Table II, our method consistently outperforms baseline SimGCD on fine-grained datasets. Specifically, on CUB and Stanford Cars, SimGCD+Debiasing achieves improvements of **7.0%/8.9%** and **9.1%/9.9%** on the *Old* and *New* metrics in DINO, and **3.2%/7.8%** and **9.9%/6.0%** in DINOv2, respectively. While our method does not surpass the latest approaches such as AptGCD, RLGCD, MOS and PCAGCD, SimGCD+Debiasing remains competitive among plug-and-play extensions of SimGCD, achieving, for example, a **3.8%** gain in DINO and a **2.4%** gain in DINOv2 on CUB compared to Hyp-SimGCD.

2) Evaluation on generic datasets: Ours DebiasGCD (SimGCD+Debiasing) demonstrates strong performance on generic object recognition datasets, as shown in Table III and Table IV. In particular, it achieves notable improvements over SimGCD on the *All* metric: **4.1%** on CIFAR100 and **2.0%** on ImageNet-100 in DINO, and **5.0%/1.3%** in DINOv2. Although DebiasGCD is slightly behind AptGCD on ImageNet-100 by 2.8 points (85.0% vs. 87.8%) in DINO, it surpasses AptGCD on CIFAR100 by **1.4%** (84.2% vs. 82.8%). With DINOv2, DebiasGCD also remains competitive with other plug-and-play extensions such as Hyp-SimGCD, achieving an additional **2.0%** gain on the *All* metric. On more challenging datasets—Herbarium19, CIFAR-LT, and ImageNet-1K—DebiasGCD achieves further gains on the *All* metric, outperforming SimGCD by **1.2%** on Herbarium19, by **7.3%** on ImageNet-1K, and by **3.1%/2.2%/2.1%** under 20/50/80 imbalance factors on CIFAR100, thereby demonstrating its robustness and competitiveness on complex benchmarks.

TABLE IV: Results on more challenging datasets with DINO.

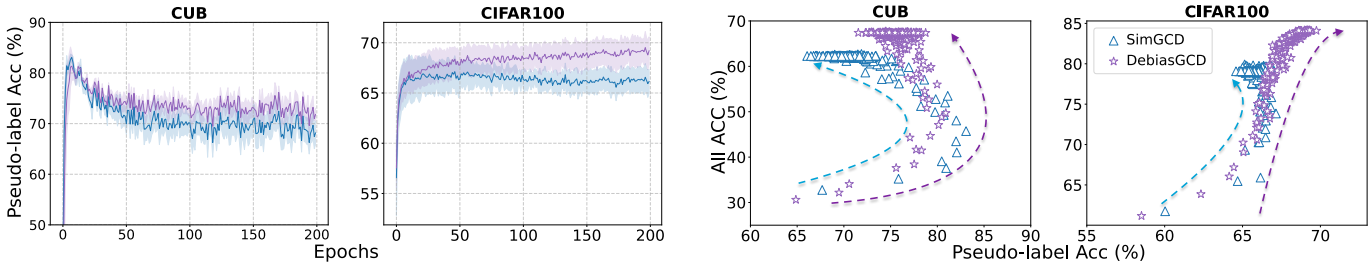
Method	Herbarium			ImageNet-1K		
	All	Old	New	All	Old	New
k-means [64]	13.0	12.2	13.4	-	-	-
RS+ [51]	27.9	55.8	12.8	-	-	-
UNO+ [50]	28.3	53.7	14.7	-	-	-
ORCA [65]	20.9	30.9	15.5	-	-	-
GCD [12]	35.4	51.0	27.0	52.5	72.5	42.2
PromptCAL [40]	37.0	52.0	28.9	-	-	-
SPTNet [16]	43.4	58.7	35.2	-	-	-
LegoGCD [17]	45.1	57.4	38.5	62.4	79.5	53.8
CMS [66]	36.4	54.9	26.4	-	-	-
RLGCD [68]	**46.4**	**61.2**	38.4	-	-	-
PALGCD [70]	46.8	57.0	**41.4**	-	-	-
SimGCD [15]	44.0	58.0	36.4	57.1	77.3	46.9
SimGCD+AF [72]	45.5	59.0	38.3	-	-	-
SimGCD+Debiasing (Ours)	45.2	60.1	37.0	**62.8**	**79.8**	**54.1**

TABLE V: Results on CIFAR-LT under imbalance factors 20, 50, and 80 [61].

Method	CIFAR-LT								
	20			50			80		
	All	Old	New	All	Old	New	All	Old	New
SimGCD	60.4	62.8	48.0	56.1	58.5	37.3	34.8	52.8	20.1
Ours	63.5	65.6	51.0	58.9	61.7	40.0	36.9	54.4	22.4

3) Pseudo-label assignments during training: We visualize the evolution of pseudo-labeling (over random seeds) for seen classes across training epochs on both fine-grained and generic datasets. As shown in Fig. 3a, our debiasing strategy consistently improves pseudo-label quality—for instance, accuracy on CUB improves from 68% to 72%, and on CIFAR100, it mitigates the decline in SimGCD, increasing from 65% to 67%. Fig. 4 further shows the pseudo-label assignment distributions, where darker points indicate higher frequencies. In particular, the red rectangles highlight regions where seen-class samples (y-axis) are mislabeled as novel classes (x-axis). Intuitively, Our DebiasGCD clearly lightens these areas, indicating reduced bias and more balanced pseudo-labeling. Moreover, Fig. 3b illustrates that higher pseudo-label accuracy generally leads to higher *All* ACC, showing a positive correlation across both fine-grained (CUB) and generic (CIFAR100) datasets.

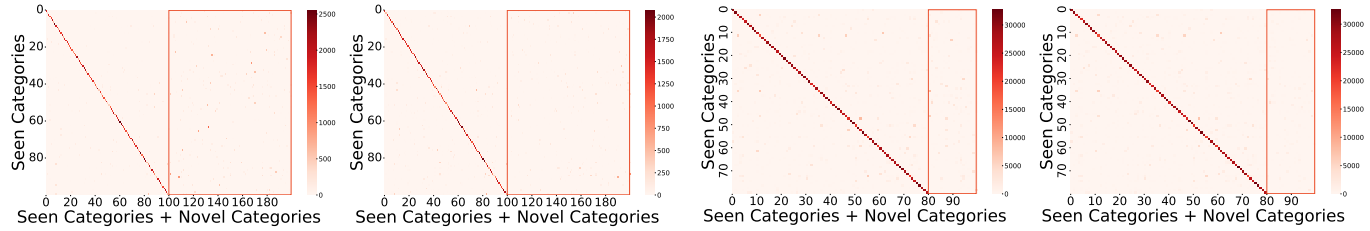
4) Feature Visualizations and Attention Distributions: Fig. 6a uses t-SNE to visualize the feature distribution of five randomly selected CIFAR100 categories, including both seen and novel classes. Intuitively, SimGCD shows overlapping novel features (red circle), suggesting poor separation, while



(a) Pseudo-label accuracy for seen classes in the unlabeled data.

(b) Variation of *All* ACC with pseudo-label accuracy.

Fig. 3: (a) Pseudo-label accuracy over three random seeds (solid lines: mean, shaded region: std). DebiasGCD (purple) consistently outperforms SimGCD (blue). (b) *All* ACC versus pseudo-label accuracy (△: SimGCD, ⋆: DebiasGCD). On CUB, both methods exhibit a forgetting effect—pseudo-label accuracy rises and then falls—but DebiasGCD is more stable (75–80% vs. 70–75%), resulting in higher *All* ACC. On CIFAR100, *All* ACC increases as pseudo-label accuracy improves: SimGCD plateaus at 65–73% pseudo-label accuracy (with *All* ACC of 80%), while DebiasGCD reaches about 70% pseudo-labels and achieves 84% *All* ACC.



(a) Group 1: CUB Dataset Results

(b) Group 2: CIFAR100 Dataset Results

Fig. 4: Pseudo-labeling in SimGCD (Column 1, with bias) and our method (Column 2, with less noise) for CUB and CIFAR100. Our DebiasGCD shows lighter colors, indicating reduced imbalanced pseudo-labeling. (Note that there are 80 seen categories and 20 novel categories in CIFAR100.).

our method forms clearer boundaries and more distinct clusters. Furthermore, Fig. 5 presents attention maps of different heads from image encoder Φ, highlighting their focus on various image regions. Compared to SimGCD, our method more effectively captures class-specific object parts and suppresses background noise, highlighting its advantage in local representation learning.

5) Computational cost comparison: We compare the training and inference costs of DebiasGCD with SimGCD and other competitive methods, including MOS [69], PALGCD [70], and SimGCD+AF [72] , in Table VI. Compared with the baseline SimGCD, DebiasGCD incurs minimal additional overhead. On CUB, it adds only 3 minutes of training while slightly reducing FLOPs (33.74 vs. 36.36). On CIFAR100, it adds 2.5 hours with moderate increases in FLOPs and parameters. Our SimGCD+Debiasing outperforms MOS and PALGCD in training efficiency (4h vs. 9h and 6h, respectively). Meanwhile, SimGCD+AF substantially increases the number of parameters (144.92 vs. 92.10). The large gap in training time across datasets (4h 35m vs. 31h 13m) is primarily due to dataset size (6.1k vs. 20.0k samples) rather than the SARD module. Overall, our approach is computationally efficient, introducing minimal training overhead and no additional inference cost.

TABLE VI: Comparison of computational cost on CUB and CIFAR100 across competitive methods.

Method	Dataset	Time		FLOPs	Params (M)
		Train	Infer		
SimGCD	CUB	4h 35m	19s	33.74	92.10
RLGCD		4h 38m	19s	33.74	92.10
MOS		9h 12m	63s	67.48	93.82
PALGCD		6h 41m	19s	33.74	92.10
SimGCD+AF		4h 36m	20s	33.85	144.92
SimGCD+Debiasing (Ours)		4h 38m	20s	36.36	98.79
SimGCD	CIFAR100	31h 13m	112s	33.82	92.02
RLGCD		31h 15m	112s	33.82	92.02
PALGCD		50h 19m	112s	33.82	92.02
SimGCD+AF		32h 21m	121s	33.96	144.68
SimGCD+Debiasing (Ours)		33h 51m	124s	35.14	98.50

C. Ablation Study

1) Effect of Strong/Weak Augmentation: We apply *strong* augmentation to the student network and *weak* augmentation to the teacher network. As discussed in [18], this strategy helps the teacher network produce more stable pseudo-labels, motivating us to introduce stable debiasing for pseudo-labeling. Row (1) of Table VII shows that this strategy alone results in only a modest improvement of 0.6% and 0.9% in 'All' accuracy on the CUB and Stanford Cars datasets, respectively. However, when combined with DPD and SARD (Rows (4) *vs*

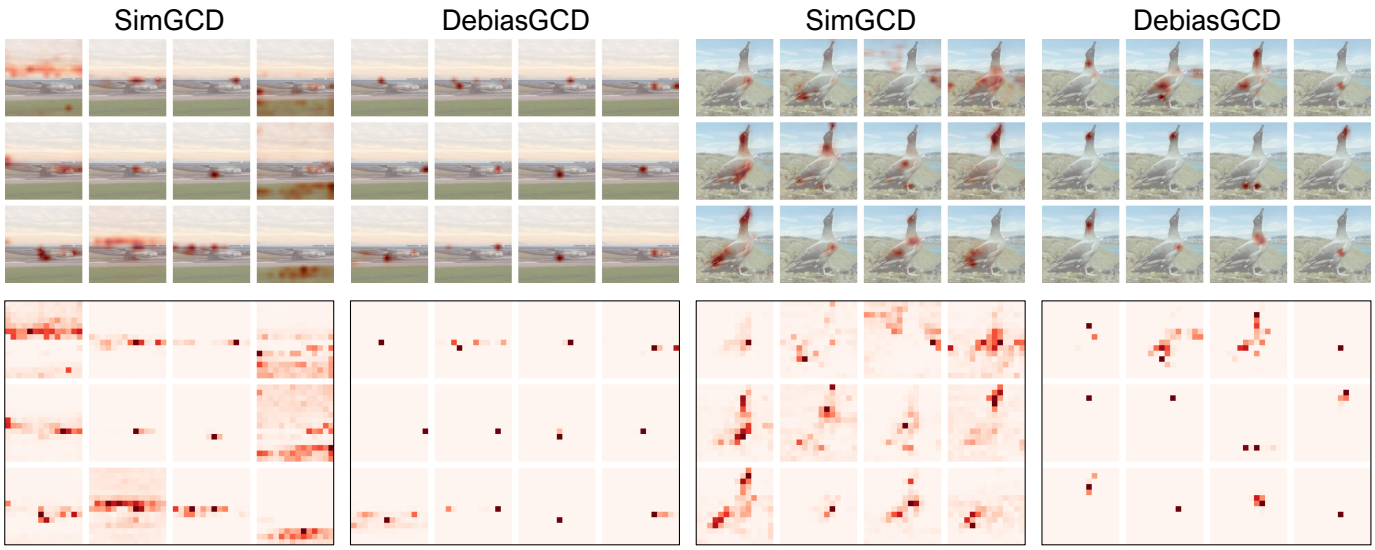
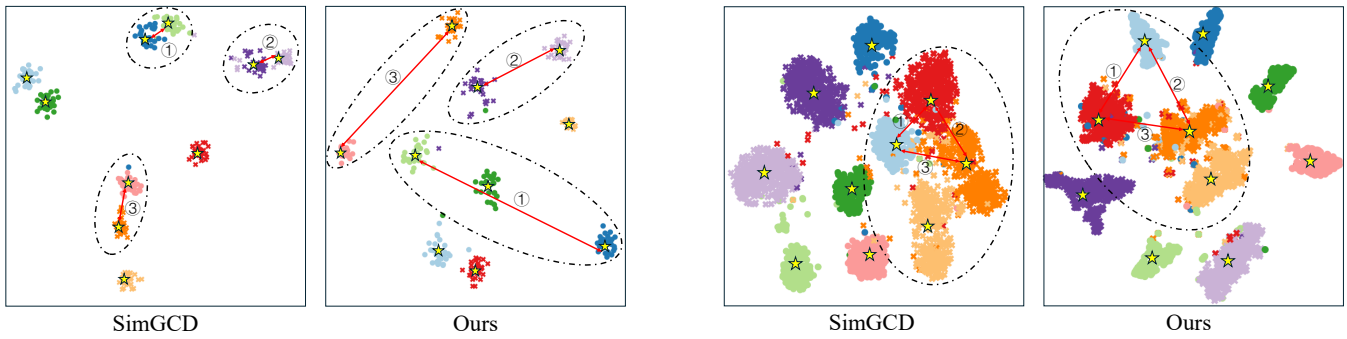


Fig. 5: Attention visualization of 12 different attention heads in the final layer of the image encoder on FGVC-Aircraft and CUB. Compared to SimGCD, our approach, DebiasGCD, focuses more on class-specific features while reducing background interference, leading to more refined and discriminative feature representations.



(a) CUB [11] (b) CIFAR100 [57]

Fig. 6: t-SNE visualization of SimGCD [15] and our DebiasGCD on 10 randomly sampled classes from CUB and CIFAR-100 datasets. ● denotes seen classes, ✖ denotes novel classes, and the five-pointed star indicates the prototype center. Compared with SimGCD, DebiasGCD yields fewer misclassified samples and clearer margins between prototypes.

TABLE VII: Effect of each component of our method. Row (0) shows the baseline results.

No.	S/W Aug.	DPD	SARD	CUB			Stanford Cars			CIFAR100		
				All	Old	New	All	Old	New	All	Old	New
(0)	✗	✗	✗	60.3	65.6	57.7	53.8	71.9	45.0	80.1	81.2	77.8
(1)	✓			60.9	68.7	57.0	54.9	74.1	45.6	80.9	81.9	79.1
(2)		✓		66.0	72.3	62.9	57.8	77.4	47.6	82.3	82.7	81.8
(3)			✓	65.3	72.1	61.9	58.0	77.0	48.9	82.1	82.6	79.4
(4)		✓	✓	67.5	**73.7**	64.5	62.4	79.7	54.1	83.0	83.8	82.7
(5)	✓	✓	✓	**68.6**	72.6	**66.6**	**63.4**	**80.1**	**54.9**	**84.2**	**84.3**	**84.0**

(5)), it leads to a substantial performance boost of 1.1% and 1.0%. These results highlight that integrating the augmentation strategy with our approach improves the stability of debiasing and enhances the quality of pseudo-labels.

2) Effect of DPD and SARD: Table VII presents the performance impact of different components. Incorporating DPD (Row (0) *vs* Row (2)) improves all metrics for CUB and Stanford Cars and CIFAR100, achieving gains of 5.7%, 4.0% and 2.2% in 'All' accuracy, respectively. This demonstrates DPD's ability to generate more discriminative prototypes for pseudo-labeling. Similarly, incorporating SARD (Row (0) *vs* Row (3)) enhances performance across both 'Old' and 'New' classes compared to SimGCD, highlighting the importance of local spatial-aware representation learning. Furthermore, combining SARD with DPD leads to additional improvements in 'All' accuracy by 1.5%, 4.6% and 0.7% (Rows (2) *vs* (4)), confirming that SARD reinforces the debiasing effect of DPD by leveraging local feature information.

3) Effect of N in DPD: Fig. 7 presents the performance trends for different N values across all datasets. Here, $N = 0$ corresponds to baseline SimGCD without dynamic prototype debiasing. When $N = 1$, the smallest value in the probability vector $\boldsymbol{p}^{DPD}(\boldsymbol{x}_S)$ is selected, resulting in $p_{N-\min} = p_1$ in Eq. (10). Similarly, for $N = 2$, the second smallest value is chosen, yielding $p_{N-\min} = p_2$. Compared to the baseline ($N = 0$), our method consistently enhances performance across all metrics ('All', 'Old', and 'New') for various N

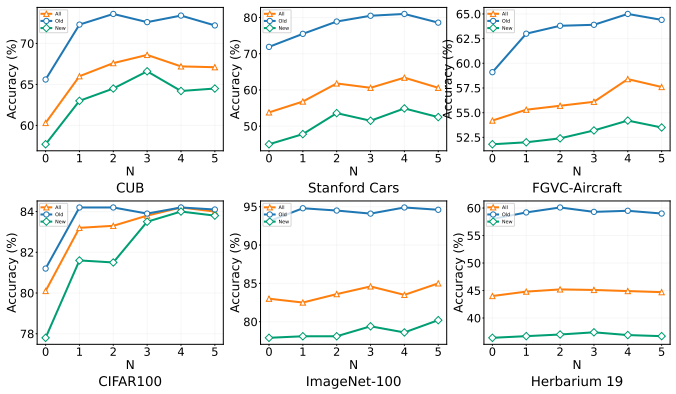


Fig. 7: Effect of varying N-th parameter in DPD. Each figure presents the accuracy of the 'All' metric across different values of N, ranging from 0 to 5.

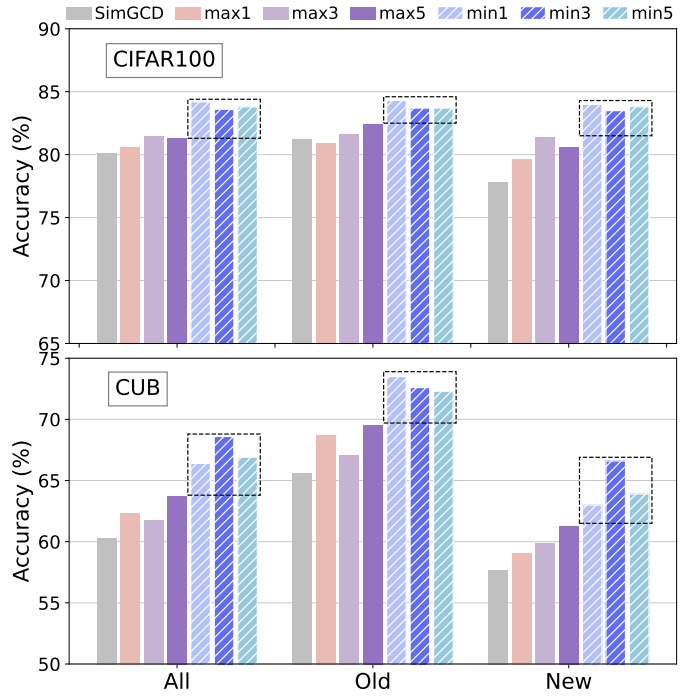


Fig. 8: Comparison of using the margin between the ground-truth prediction and either $p_{N-\min}$ or $p_{N-\max}$ in Eq. (10). Using $p_{N-\min}$ consistently outperforms $p_{N-\max}$ for $N \in \{1, 3, 5\}$.

values across all datasets. Notably, the improvement is most pronounced in CUB, Stanford Cars, and CIFAR100. Meanwhile, we investigate the impact of choosing $p_{N-\min}$ versus $p_{N-\max}$. As shown in Fig. 8, the first bar denotes the baseline SimGCD; bars 2–4 correspond to $p_{N-\max}$ with $N \in \{1, 3, 5\}$, and bars 5–7 correspond to $p_{N-\min}$ with $N \in \{1, 3, 5\}$. The black rectangles mark the performance gains over $p_{N-\max}$. Overall, $p_{N-\min}$ consistently outperforms $p_{N-\max}$ in all cases.

4) Effect of debiasing margin ϵ: Table VIII illustrates the effect of varying the margin ϵ (0.1, 0.3, 0.5, 1, and 2) in Eq. (10). Specifically, the margin determines the distance between the predicted probability p_y for the ground truth label and the probability of the N-th smallest value $p_{N-\min}$. As ϵ increases from 0.1 to 2, performance improves across all datasets. However, when ϵ further increases from 1 to 2, performance declines in Stanford Cars and CIFAR100. This suggests that while a small margin fails to capture discriminative features, an excessively large margin disrupts shared class features, ultimately hindering performance.

TABLE VIII: Effect of varying margin ϵ of Eq. (10) in both fine-grained and generic object recognition datasets.

ϵ	CUB			Stanford Cars			CIFAR100		
	All	Old	New	All	Old	New	All	Old	New
0.1	60.1	65.3	57.5	53.7	71.5	45.2	80.0	81.5	77.5
0.1	61.6	68.3	58.5	55.6	72.7	47.5	81.6	82.3	79.7
0.3	63.9	71.5	60.6	58.6	74.5	51.5	82.2	82.6	81.4
0.5	65.1	73.1	61.6	61.8	78.9	53.6	83.8	83.9	83.5
1.0	67.4	**76.3**	63.0	**63.4**	**81.0**	**54.9**	**84.2**	**84.2**	**84.0**
2.0	**68.8**	72.6	**66.6**	62.4	79.7	54.0	84.0	84.1	83.8

5) Effect of Hyperparameters α and β: Table IX shows the values of α and β used in Eq. (14). The parameter α controls the impact of detailed feature learning on the prototype classifier, while β balances the prototype margins within DPD. In fine-grained datasets like CUB and Aircraft, performance 'All' improves as α increases from 0.5 to 1, highlighting the importance of local features for fine-grained recognition. For a generic dataset CIFAR100, the best performance is achieved when both α and β are set to 0.5. Although performance in CIFAR100 decreases as α and β increase, it still outperforms the baseline SimGCD ('All'=80.1).

TABLE IX: Ablation study on α and β of Eq. (14) in fine-grained and generic object recognition datasets.

α	β	CUB			FGVC-Aircraft			CIFAR100		
		All	Old	New	All	Old	New	All	Old	New
0.5	0.5	66.0	73.9	62.1	56.1	65.1	52.3	**84.2**	84.2	**84.0**
	1.0	67.5	**74.6**	63.9	55.8	65.1	51.2	83.8	83.9	83.5
1	0.5	67.2	73.3	64.1	56.3	64.8	52.1	83.2	84.2	81.6
	1.0	**68.6**	72.6	**66.6**	**56.8**	**65.7**	**52.4**	82.6	**84.4**	78.9

6) Effective Debiasing for other GCD models: To further examine the generalization ability of DPD, we incorporated it into LegoGCD [17] and SPTNet [16], both built upon SimGCD. Notably, we set the hyperparameters for CUB and CIFAR100 following the implementation details in Section V-A3. As shown in Table X, LegoGCD+DPD improves over LegoGCD by +2.9% on CUB (All) and +2.7% on CIFAR (All), while SPTNet+DPD achieves gains of +1.5% and +1.3%, respectively. These results indicate that pseudo-label bias is a common challenge in GCD models, and DPD provides a simple yet effective solution that generalizes well across different frameworks.

VI. Conclusion

In this paper, we find the previously unaware bias of imbalanced pseudo-labeling in the GCD task caused by prototypes without clear margins. To mitigate this bias, we propose an effective debiasing method, DebiasGCD, which adopts margin learning between seen and novel category prototypes.

TABLE X: Results of incorporating DPD into two additional methods: LegoGCD [17] and SPTNet [16].

Datasets	CUB			CIFAR100		
	All	Old	New	All	Old	New
LegoGCD	63.8	71.9	59.8	81.8	81.4	82.5
LegoGCD+DPD	66.7	73.1	63.7	83.5	83.2	83.7
SPTNet	65.8	68.8	65.1	81.3	84.3	75.6
SPTNet+DPD	67.3	70.1	66.4	82.6	84.5	79.1

Specifically, we propose a dynamic prototype debiasing technique to maintain a margin between prototypes dynamically, encouraging the network to explore category-specific features and enhance prototype distinction. Furthermore, to improve the learning for more discriminative representations in DPD, we propose a spatial-aware representations distillation module to discover more subtle features that benefit classification, especially in fine-grained datasets. Extensive results show that our DebiasGCD significantly outperforms the baseline SimGCD, effectively mitigating the pseudo-labeling bias.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[4] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *CVPR*, 2021.

[5] H. Yue, J. Guo, X. Yin, Y. Zhang, B. Wen, and C. Li, "Salient object detection toward single-pixel imaging," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 1, pp. 235–247, 2024.

[6] L. Ding, X. Song, Y. He, C. Wang, S. Dong, X. Wei, and Y. Gong, "Domain incremental object detection based on feature space topology preserving strategy," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 1, pp. 424–437, 2024.

[7] D. Berthelot, N. Carlini, I. J. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *NeurIPS*, 2019.

[8] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. Raffel, E. D. Cubuk, A. Kurakin, and C. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," in *NeurIPS*, 2020.

[9] L. Ran, Y. Li, G. Liang, and Y. Zhang, "Pseudo labeling methods for semi-supervised semantic segmentation: A review and future perspectives," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 35, no. 4, pp. 3054–3080, 2025.

[10] Y. Tang, Z. Cao, Y. Yang, J. Liu, and J. Yu, "Semi-supervised few-shot object detection via adaptive pseudo labeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 4, pp. 2151–2165, 2024.

[11] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[12] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Generalized category discovery," in *CVPR*, 2022.

[13] B. Zhao, X. Wen, and K. Han, "Learning semi-supervised gaussian mixture models for generalized category discovery," in *ICCV*, 2023.

[14] N. Pu, Z. Zhong, and N. Sebe, "Dynamic conceptional contrastive learning for generalized category discovery," in *CVPR*, 2023.

[15] X. Wen, B. Zhao, and X. Qi, "Parametric classification for generalized category discovery: A baseline study," in *ICCV*, 2023.

[16] H. Wang, S. Vaze, and K. Han, "Sptnet: An efficient alternative framework for generalized category discovery with spatial prompt tuning," in *ICLR*, 2024.

[17] X. Cao, X. Zheng, G. Wang, W. Yu, Y. Shen, K. Li, Y. Lu, and Y. Tian, "Solving the catastrophic forgetting problem in generalized category discovery," in *CVPR*, 2024.

[18] S. Vaze, A. Vedaldi, and A. Zisserman, "No representation rules them all in category discovery," in *NeurIPS*, 2023.

[19] M. Assran, M. Caron, I. Misra, P. Bojanowski, F. Bordes, P. Vincent, A. Joulin, M. Rabbat, and N. Ballas, "Masked siamese networks for label-efficient learning," in *ECCV*, 2022.

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS*, 2014.

[21] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *ICLR*, 2017.

[22] Z. Huang, C. Xue, B. Han, J. Yang, and C. Gong, "Universal semi-supervised learning," in *NeurIPS*, 2021.

[23] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, "Long-tail learning via logit adjustment," in *ICLR*, 2021.

[24] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *NeurIPS*, 2017.

[25] T. Miyato, S. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *TPAMI*, 2019.

[26] Q. Xie, Z. Dai, E. H. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *NeurIPS*, 2020.

[27] W. Zhang, L. Zhu, J. Hallinan, S. Zhang, A. Makmur, Q. Cai, and B. C. Ooi, "Boostmis: Boosting medical image semi-supervised learning with adaptive pseudo labeling and informative active annotation," in *CVPR*, 2022.

[28] Z. Yu, Y. Li, and Y. J. Lee, "Inpl: Pseudo-labeling the inliers first for imbalanced semi-supervised learning," in *ICLR*, 2023.

[29] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *NeurIPS*, 2020.

[30] J. Wu, H. Fan, Z. Li, G. Liu, and S. Lin, "Information transfer in semi-supervised semantic segmentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 2, pp. 1174–1185, 2024.

[31] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, "Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring," in *ICLR*, 2020.

[32] Y. Wang, H. Chen, Q. Heng, W. Hou, Y. Fan, Z. Wu, J. Wang, M. Savvides, T. Shinozaki, B. Raj, B. Schiele, and X. Xie, "Freematch: Self-adaptive thresholding for semi-supervised learning," in *ICLR*, 2023.

[33] Y. Xu, L. Shang, J. Ye, Q. Qian, Y. Li, B. Sun, H. Li, and R. Jin, "Dash: Semi-supervised learning with dynamic thresholding," in *ICML*, M. Meila and T. Zhang, Eds., 2021.

[34] B. Zhang, Y. Wang, W. Hou, H. Wu, J. Wang, M. Okumura, and T. Shinozaki, "Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling," in *NeurIPS*, 2021.

[35] Z. Huang, L. Shen, J. Yu, B. Han, and T. Liu, "Flatmatch: Bridging labeled data and unlabeled data with cross-sharpness for semi-supervised learning," in *NeurIPS*, 2023.

[36] P. Du, S. Zhao, Z. Sheng, C. Li, and H. Chen, "Semi-supervised learning via weight-aware distillation under class distribution mismatch," in *ICCV*, 2023.

[37] H. Zheng, Q. Wang, Z. Fang, X. Xia, F. Liu, T. Liu, and B. Han, "Out-of-distribution detection learning with unreliable out-of-distribution sources," in *NeurIPS*, 2023.

[38] Z. Sun, Y. Qiu, Z. Tan, W. Zheng, and R. Wang, "Classifier-head informed feature masking and prototype-based logit smoothing for out-of-distribution detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 7, pp. 5630–5640, 2024.

[39] Z. Peng, E. Wang, X. Liu, and M. Cheng, "Predictive sample assignment for semantically coherent out-of-distribution detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 35, no. 5, pp. 4686–4697, 2025.

[40] S. Zhang, S. H. Khan, Z. Shen, M. Naseer, G. Chen, and F. S. Khan, "Promptcal: Contrastive affinity learning via auxiliary prompts for generalized novel category discovery," in *CVPR*, 2023.

[41] Y. Fei, Z. Zhao, S. Yang, and B. Zhao, "Xcon: Learning with experts for fine-grained category discovery," in *BMVC*, 2022, p. 96.

[42] E. Wang, Z. Peng, Z. Xie, H. Lu, F. Yang, and X. Liu, "Learning part knowledge to facilitate category understanding for fine-grained generalized category discovery," *CoRR*, vol. abs/2503.16782, 2025.

[43] E. Wang, Z. Peng, Z. Xie, F. Yang, X. Liu, and M. Cheng, "GET: unlocking the multi-modal potential of CLIP for generalized category discovery," in *CVPR*, 2025, pp. 20 296–20 306.

[44] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *ICML*, vol. 139, 2021, pp. 8748–8763.

[45] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *NeurIPS*, 2017.
[46] W. An, F. Tian, Q. Zheng, W. Ding, Q. Wang, and P. Chen, "Generalized category discovery with decoupled prototypical network," in *AAAI*, 2023.
[47] W. An, F. Tian, W. Shi, Y. Chen, Y. Wu, Q. Wang, and P. Chen, "Transfer and alignment network for generalized category discovery," in *AAAI*, 2024, pp. 10 856–10 864.
[48] H. Yang, B. Sun, B. Li, C. Yang, Z. Wang, J. Chen, L. Wang, and H. Li, "Iterative class prototype calibration for transductive zero-shot learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 3, pp. 1236–1246, 2023.
[49] Y. Hu, L. Feng, H. Jiang, M. Liu, and B. Yin, "Domain-aware prototype network for generalized zero-shot learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 5, pp. 3180–3191, 2024.
[50] E. Fini, E. Sangineto, S. Lathuilière, Z. Zhong, M. Nabi, and E. Ricci, "A unified objective for novel class discovery," in *ICCV*, 2021.
[51] K. Han, S. Rebuffi, S. Ehrhardt, A. Vedaldi, and A. Zisserman, "Autonovel: Automatically discovering and learning novel visual categories," *TPAMI*, 2022.
[52] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *NeurIPS*, 2020.
[53] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.
[54] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, 2018.
[55] X. Wang, Z. Wu, L. Lian, and S. X. Yu, "Debiased learning from naturally imbalanced pseudo-labels," in *CVPR*, 2022, pp. 14 627–14 637.
[56] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *CVPR*, 2016, pp. 2921–2929.
[57] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.
[58] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *ICCV*, 2013.
[59] S. Maji, E. Rahtu, J. Kannala, M. B. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013.
[60] K. C. Tan, Y. Liu, B. Ambrose, M. Tulig, and S. Belongie, "The herbarium challenge 2019 dataset," *arXiv preprint arXiv:1906.05372*, 2019.
[61] K. Cao, C. Wei, A. Gaidon, N. Aréchiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *NeurIPS*, 2019, pp. 1565–1576.
[62] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, 1955.
[63] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P. Huang, S. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "Dinov2: Learning robust visual features without supervision," *TMLR*, vol. 2024, 2024.
[64] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1967.
[65] J. Liu, Y. Wang, T. Zhang, Y. Fan, Q. Yang, and J. Shao, "Open-world semi-supervised novel class discovery," in *IJCAI*, 2023.
[66] S. Choi, D. Kang, and M. Cho, "Contrastive mean-shift learning for generalized category discovery," in *CVPR*, 2024, pp. 23 094–23 104.
[67] W. Zhang, B. Zhang, Z. Teng, W. Luo, J. Zou, and J. Fan, "Less attention is more: Prompt transformer for generalized category discovery," in *CVPR*, 2025, pp. 30 322–30 331.
[68] D. Liu, Z. Tan, L. Zhao, Z. Zhang, X. Fang, and W. Huang, "Generalized category discovery via reciprocal learning and class-wise distribution regularization," *ICML*, 2025.
[69] Z. Peng, J. Ma, Z. Sun, R. Yi, H. Song, X. Tan, and L. Ma, "MOS: modeling object-scene associations in generalized category discovery," in *CVPR*, 2025, pp. 15 118–15 128.
[70] M. Wang, Z. Zhong, and X. Gong, "Prior-constrained association learning for fine-grained generalized category discovery," in *AAAI*, 2025, pp. 21 162–21 170.
[71] Y. Liu, Z. He, and K. Han, "Hyperbolic category discovery," in *CVPR*, 2025, pp. 9891–9900.
[72] Q. Xu, Z. Hu, Y. Duan, E. Pei, and Y. Tai, "A hidden stumbling block in generalized category discovery: Distracted attention," *ICCV*, 2025.
[73] S. Hao, K. Han, and K. K. Wong, "Cipr: An efficient framework with cross-instance positive relations for generalized category discovery," *TMLR*, 2024.

Xinzi Cao received the M.S. degree from the School of Computer Science and Engineering, Sun Yat-sen University, in 2022. She is currently a Ph.D. candidate jointly supervised by the School of Computer Science and Engineering at Sun Yat-sen University and the Peng Cheng Laboratory in Shenzhen, China. Her research interests include open-set semi-supervised learning and category discovery.

Feidiao Yang is currently an associate researcher at Pengcheng Laboratory (PCL). Before joining PCL, he worked as a researcher at Microsoft Research AI for Science (MSR-AI4S) and Microsoft Research Asia (MSRA). He got his Ph.D. degree from Institute of Computing Technology, Chinese Academy of Sciences, supervised by Professor Xiaoming SUN. His research interest mainly includes algorithm, software, and application of machine learning and artificial intelligence, especially their inter discipline research with other fields.

Xiawu Zheng received his Ph.D. degree in Computer Science in 2020 from the School of Information Science and Engineering, Xiamen University, Xiamen, China. He is currently a professor at the School of Informatics, Xiamen University. His research interests include computer vision and machine learning, with a particular focus on automated machine learning (AutoML).

Quanmin Liang received the B.S. degree from the School of Computer Science, Sun Yat-sen University, Guangzhou, China, in 2018, and the M.S. degree from the School of Psychology, Sun Yat-sen University, in 2022. He is currently pursuing the Ph.D. degree with the School of Computer Science, Sun Yat-sen University. His research interests include computer vision and machine learning.

Yutong Lu is a professor at Sun Yat-sen University and director at the National Super-computing Center in Guangzhou, specializes in Chinese supercomputing generations. Her research spans HPC, Cloud computing, storage and programming. She currently dedicates effort to converging HPC, Big Data, and AI on supercomputers for system and application implementation.

Yonghong Tian is currently a Boya Distinguished Professor with the School of Computer Science, Peking University, China, and the Deputy Director of the Peng Cheng Laboratory, Artificial Intelligence Research Center, Shenzhen, China. His research interests include neuromorphic vision, distributed machine learning, and multimedia big data. He is the author or coauthor of over 280 technical papers in refereed journals and conferences. He was/is an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.