Propagate to Discover: Graph-Structured Propagation for Generalized Category Discovery

Junyong Liu[1,*], Yongqi Tian[2,*], Jinkun Ran[2], Haoyuan He[2], Caigui Jiang[2]
[1]School of Information and Communication Engineering,
Beijing University of Posts and Telecommunications
[2]State Key Laboratory of Human-Machine Hybrid Augmented Intelligence,
Institute of Artificial Intelligence and Robotics,
Xi'an Jiaotong University
1536308188@qq.com, yongqitian@stu.xjtu.edu.cn, jinkunran@163.com
hehy02@stu.xjtu.edu.cn, cgjiang@xjtu.edu.cn
[*]Equal contribution.

Abstract

Generalized category discovery learns to recognize old and novel classes from partially labeled data. In its active setting, a limited annotation budget can resolve ambiguities among categories that have no initial labels. The value of each query also depends on how its label reaches the remaining unlabeled samples. We propose PD-GCD, a framework that connects active label acquisition with graph propagation and representation learning. Anchor Mining selects representative and informative samples for annotation. Their acquired labels, together with the initial labels, seed a neighborhood graph over the training data. During active querying, propagated uncertainty guides the selection of further samples. After the query budget is spent, graph assignments supervise a parametric model through distillation and graph consistency. Updated representations then refine the graph used for supervision. This interaction lets acquired labels influence both nearby samples and the representation used for discovery. Experiments cover six generic and fine-grained benchmarks. Under this active annotation setting, PD-GCD reaches New accuracy of 74.9% on CUB and 75.5% on Stanford Cars. The mechanism study shows that graph propagation is most effective when paired with informative labeled anchors.

1 Introduction

Large-scale pretraining has advanced visual representation learning [1] and few-shot language learning [2]. Recognition systems often encounter categories that were absent from their labeled training set, motivating open-world and open-set recognition [3, 4]. Generalized category discovery (GCD) studies this setting using labeled examples from old categories and an unlabeled pool that contains both old and novel categories [5]. The model must preserve distinctions it already knows while learning new ones from the same pool. Recent methods improve this process through parametric classification, contrastive learning, and prototype learning [6, 7, 8]. Fine-grained discovery remains difficult because neighboring categories can share most of their visual appearance. Labels from old categories help shape the feature space, but they may provide little guidance about the details that separate two novel categories.

Active GCD addresses this ambiguity by allowing the model to request a limited number of additional labels [9]. A queried image may belong to an old or a novel category, so each answer can improve recognition or establish a previously missing category distinction. PD-GCD follows this active setting and its cycle of querying and retraining. The central question is how to make each acquired label useful beyond the queried image. A label can provide evidence for many unlabeled

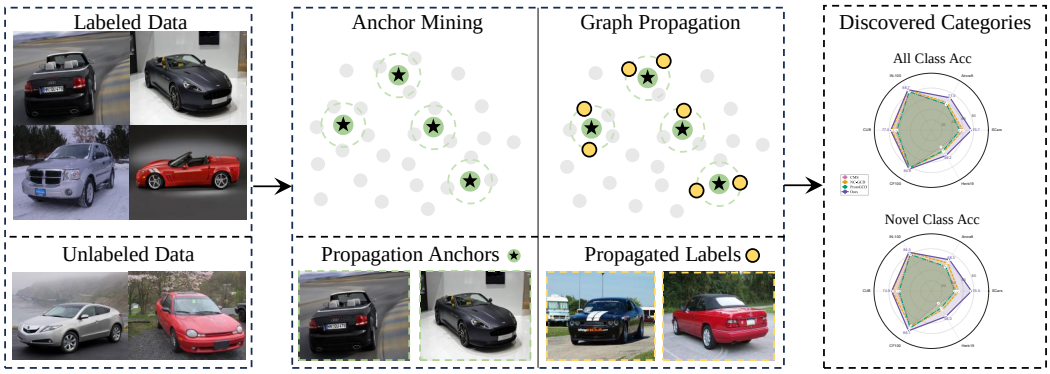
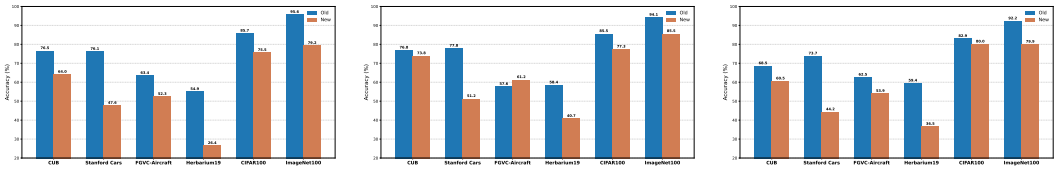


Figure 1: Overview of PD-GCD. Anchor Mining selects samples from the unlabeled pool for annotation. The acquired labels seed Graph Propagation, which produces soft supervision for the remaining training samples. Graph uncertainty guides the active queries. After querying, distillation updates the representation and the graph used for refinement.



(a) CMS[7] (b) NC-GCD[13] (c) ProtoGCD[8]

Figure 2: Old and New accuracy of three GCD methods on six benchmarks. The gap is pronounced on several fine-grained datasets, motivating additional supervision for ambiguous novel categories.

neighbors, but its reach depends on which regions are sampled and how accurately the current representation connects them.

This dependence is important when old and novel categories overlap in feature space. Selecting only typical examples can leave ambiguous regions without supervision. Selecting only uncertain examples can spend much of the budget on visually similar points. After annotation, a graph offers a way to distribute the acquired information across the pool [10, 11, 12]. Its connections are also imperfect. A graph built from features that mix two categories can spread the same mistake across many samples. We therefore connect label acquisition, propagation, and representation learning. Informative labels improve the graph's supervision, and a better representation changes where that supervision can travel.

We introduce PD-GCD to implement this interaction. The first stage learns a GCD representation and performs Anchor Mining to select examples for annotation. It combines samples close to cluster centers with samples in less well-covered regions. The second stage builds a neighborhood graph over the training data and uses the initial and acquired labels as semantic seeds. Graph Propagation turns these sparse labels into soft assignments for unlabeled samples. Classwise confidence filtering expands the seed support during graph propagation. During the active phase, uncertainty in the propagated assignments and feature diversity guide further queries. Once querying ends, graph-guided distillation transfers the assignments to a parametric model. A consistency loss relates feature similarities to the propagated category distributions. Rebuilding the graph from updated features then refines the supervision used in later training rounds.

We evaluate PD-GCD on six benchmarks spanning generic and fine-grained recognition. The main results show strong novel-category recognition under the active annotation setting, including 74.9% New accuracy on CUB and 75.5% on Stanford Cars. The mechanism study also reveals a concrete limitation of propagation alone: its benefit depends on the labeled anchors that support it. Our contributions are:

- We connect label acquisition with graph-based supervision in active GCD. Queried labels guide the assignments of unlabeled samples and the representation used for discovery.

- We develop PD-GCD with coverage-based Anchor Mining, uncertainty-guided queries, and graph-guided model refinement. These components connect sparse acquired labels to dense supervision over the training pool.
- We evaluate the framework on six benchmarks and analyze how anchor selection, propagation, and the training objectives affect old- and novel-category recognition.

2 Related Work

2.1 Generalized Category Discovery

Generalized category discovery jointly recognizes old categories and discovers novel ones in an unlabeled pool [5]. It extends novel category discovery by allowing unlabeled samples to come from either group [14, 15, 16]. Existing approaches learn this structure through contrastive objectives, neighborhood aggregation, or parametric classifiers [5, 17, 6, 18, 7]. More recent methods examine fine-grained discrimination, alignment between supervised and unsupervised objectives, and prototype bias [19, 13, 8]. Incremental GCD extends discovery to successive data arrivals [20]. These studies show that the representation and the assignment rule both affect the balance between old and novel categories. PD-GCD studies this balance in the active setting, where a limited number of queried labels can guide the structure learned from unlabeled data.

2.2 Active Learning

Active learning selects examples for annotation under a limited budget [21, 22]. Uncertainty methods target ambiguous predictions, while diversity methods seek broad coverage of the feature space [23, 24, 25, 26, 27]. Probabilistic self-explainable networks also quantify uncertainty in prototype-based predictions and explanations [28]. Hybrid strategies combine these criteria to reduce redundant queries [29, 30]. Active GCD (AGCD) [9] extends this process to an unlabeled pool containing old and novel categories. It uses adaptive sampling for novel categories and updates the model with both labeled and unlabeled data. PD-GCD shares this problem setting and the query-and-retrain procedure. Our method connects label acquisition to a graph: the acquired labels become propagation seeds, the graph supplies supervision for unlabeled samples, and propagated uncertainty guides later queries.

2.3 Graph Propagation and Structural Distillation

Graph-based semi-supervised learning transfers labels through neighborhood relations under a smoothness assumption [10, 31, 11, 32]. Dual sparse representation graphs support collaborative propagation for semi-supervised hyperspectral image classification [33]. Deep label propagation applies this principle to nearest-neighbor graphs built from learned embeddings [12]. Distillation and consistency learning provide ways to transfer soft predictions and relational information to a parametric model [34, 35, 36, 37, 38]. Geometric knowledge distillation transfers graph topology from a teacher to a student operating on a smaller or sparser graph [39]. Weak supervision can also take the form of text descriptions, as in referring image segmentation [40]. PD-GCD brings these tools into an active discovery process. Queried labels provide graph seeds for old and novel categories, and classwise confidence filtering extends their support. Distillation then trains the model on the propagated assignments. During the active phase, propagated uncertainty guides queries. After querying, representation updates and graph reconstruction refine the targets used for model training.

3 Method

3.1 Overview

PD-GCD addresses active generalized category discovery. We start with an initial labeled set $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^{N_l}$ from old categories and an unlabeled pool $\mathcal{D}_u = \{x_j\}_{j=1}^{N_u}$ containing both old and novel categories. Following the active setting of AGCD [9], the learner can query the labels of selected training samples under a fixed annotation budget. A query can reveal either an old category or a novel one. Old and New evaluation groups remain defined by the initial split. Queried samples join the labeled set and contribute to subsequent training.

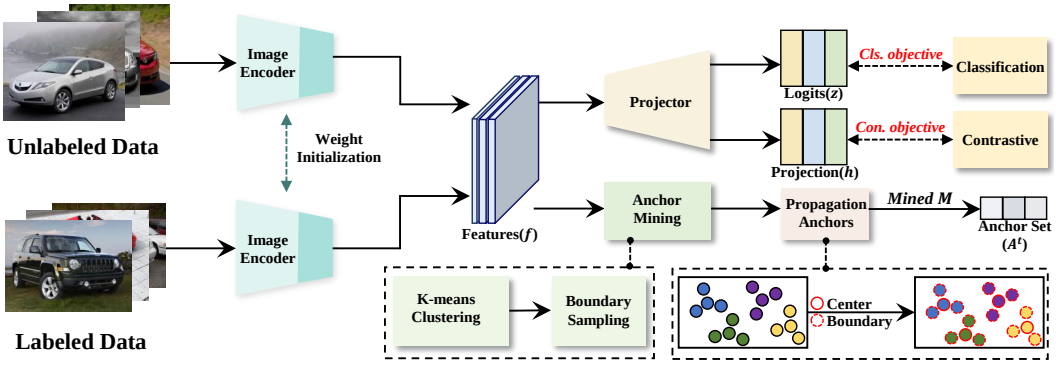


Figure 3: First-stage representation learning and Anchor Mining. Supervised and unsupervised objectives train the encoder and projector. The query strategy selects cluster representatives and then samples far from their nearest cluster center. An oracle labels the selected samples, which join the labeled training set and later provide seeds for Graph Propagation.

Figures 3 and 4 show the two stages of PD-GCD. The first stage learns an image encoder $f_\theta(\cdot)$ and a projector $g_\phi(\cdot)$ while selecting representative samples for annotation. The second stage adds an adapter $a_\psi(\cdot)$ and constructs a graph over training features. Initial and acquired labels seed propagation, which produces soft targets for model refinement and uncertainty scores for querying. Training first updates the adapter with the encoder frozen, then jointly refines the adapter and the final Transformer block. The neighborhood graph is rebuilt as the representation changes.

3.2 GCD-Driven Representation Learning

The first stage learns features that support both category discovery and sample selection. The image encoder $f_\theta(\cdot)$ maps each input to a feature vector $\mathbf{f}$. A multi-layer projector $g_\phi(\cdot)$ produces classification logits $\mathbf{z}$ and a contrastive representation $\mathbf{h}$. We use a DINO ViT-B/16 encoder for the main experiments and follow the parametric GCD training formulation of SimGCD [6].

For each input x, we generate two stochastic views $x^{(1)}$ and $x^{(2)}$ and compute

$$\mathbf{f}^{(v)} = f_\theta(x^{(v)}), \qquad (\mathbf{h}^{(v)}, \mathbf{z}^{(v)}) = g_\phi(\mathbf{f}^{(v)}), \quad v \in \{1, 2\}. \tag{1}$$

The supervised objective is $\mathcal{L}_{\text{sup}} = \mathcal{L}_{\text{supCE}} + \mathcal{L}_{\text{supCon}}$, combining classification and supervised contrastive losses on the currently labeled samples. This set includes both initial annotations and labels acquired during training. The unsupervised objective is $\mathcal{L}_{\text{unsup}} = \mathcal{L}_{\text{cluster}} + \mathcal{L}_{\text{InfoNCE}}$. The clustering term combines cross-view prediction consistency with a class-balance regularizer, while InfoNCE preserves instance discrimination. These unsupervised terms use both labeled and unlabeled samples. The combined objective is

$$\mathcal{L}_{\text{GCD}} = \lambda_{\text{sup}} \mathcal{L}_{\text{sup}} + (1 - \lambda_{\text{sup}}) \mathcal{L}_{\text{unsup}}, \tag{2}$$

where $\lambda_{\text{sup}} \in [0, 1]$ controls the balance between the supervised and unsupervised terms. Training alternates with querying, so newly acquired labels can shape the representation used for later selections.

3.3 Anchor Mining for Graph Propagation

Anchor Mining selects samples to annotate and use as graph seeds. At query round t, we extract normalized features $\mathbf{f}_i = f_\theta(x_i)$ from the remaining unlabeled pool and group them with K-means. The sampling cluster count follows the number of categories observed through the available annotations, with a small cluster count at initialization. This sampling count is distinct from the total category count used by the classifier. We first choose the sample closest to each cluster center to obtain the representative set A^t_{cen}. If the query budget remains, we select samples with the largest distance to their nearest center. This second set, A^t_{bd}, extends coverage to regions poorly represented by the selected centers. The operation labeled Boundary Sampling in Fig. 3 refers to this distance-based selection. The complete query set is

$$A^t = A^t_{\text{cen}} \cup A^t_{\text{bd}}, \qquad |A^t| = M, \tag{3}$$

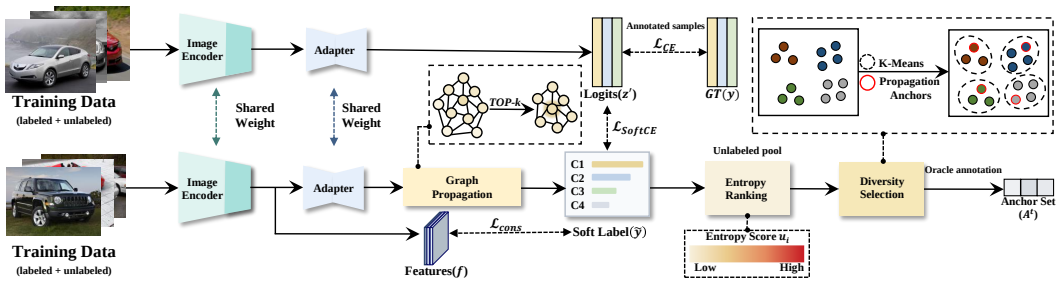


Figure 4: Second-stage graph refinement. Both branches process labeled and unlabeled training samples through a shared encoder and adapter. Initial and queried labels seed graph propagation. Hard-label cross-entropy uses initially labeled samples and samples annotated through active queries. Entropy Ranking and Diversity Selection select samples from the remaining unlabeled training pool, and an oracle annotates the selected samples. Training first updates the adapter with the encoder frozen, then jointly refines the adapter and the final encoder block. Graph construction uses normalized adapter bottleneck features, while $\mathcal{L}_{\text{cons}}$ compares normalized encoder outputs.

where M is the number of samples requested in the current query round. We obtain the ground-truth labels of these samples from the annotation oracle and add them to the labeled set. Thus, an anchor contributes an acquired label, including a novel-category label when the query discovers one. The total annotation budget counts these acquisitions across query rounds and both stages. This use of queried labels follows active GCD [9], while the selected anchors also define where supervision enters the graph.

3.4 Graph-Based Refinement via Graph Propagation and Anchor Refinement

The adapter maps encoder features $\mathbf{f}_i = f_\theta(x_i)$ to logits $\mathbf{z}'_i = a_\psi(\mathbf{f}_i)$ and an intermediate representation for graph construction. We build a k-nearest-neighbor graph from these normalized adapted features over the training set. Initial and actively acquired labels provide one-hot seeds, which are clamped during propagation. Weighted neighbor aggregation transfers these labels to the remaining nodes. We then expand the seed set with confident pseudo-labels. Within each predicted category that has an annotated seed, we retain a fraction of the most confident predictions above a confidence threshold. A second propagation pass uses the expanded seed set to produce $\tilde{\mathbf{y}}_i$. Pseudo-label expansion is recomputed from the graph and does not consume the annotation budget or add ground-truth labels to the training set.

During active querying, the encoder is frozen and the adapter learns from the currently labeled samples. Entropy Ranking selects unlabeled candidates with high entropy in their propagated class distributions. Diversity Selection clusters their features and chooses representatives for annotation. Acquired labels enter the labeled set and seed subsequent propagation. After querying, adapter-only distillation uses both annotated labels and graph soft targets. Joint refinement then updates the adapter and the final Transformer block using classification, distillation and graph consistency losses. The final encoder normalization layer remains frozen throughout the second stage. The appendix specifies the training schedules.

Model refinement combines direct supervision with graph supervision. The classification loss $\mathcal{L}_{\text{CE}}$ uses every currently annotated sample, including queried anchors. Let $p_i = \text{softmax}(\mathbf{z}'_i)$ denote the model prediction. Graph targets are sharpened with temperature T:

$$\hat{y}_{i,c} = \frac{\tilde{y}_{i,c}^{1/T}}{\sum_{c'=1}^{C} \tilde{y}_{i,c'}^{1/T}}, \tag{4}$$

where C is the classifier output dimension. The graph distillation loss is

$$\mathcal{L}_{\text{softCE}} = -\frac{1}{|\mathcal{P}^t|} \sum_{x_i \in \mathcal{P}^t} \sum_{c=1}^{C} \hat{y}_{i,c} \log p_{i,c}, \tag{5}$$

where $\mathcal{P}^t$ is the set of training samples retained for distillation. During adapter-only distillation, training uses all training samples with $T = 1$. During joint refinement, encoder fine-tuning uses

Table 1: DINOv1 results for PD-GCD and published GCD methods. PD-GCD uses budgeted label queries in the active GCD setting.

Method	CUB			Stanford Cars			FGVC Aircraft			Herbarium19			CIFAR100			ImageNet100		
	All	Old	New	All	Old	New	All	Old	New	All	Old	New	All	Old	New	All	Old	New
GCD (CVPR22)	51.3	56.6	48.7	39.0	57.6	29.9	45.0	41.1	46.9	35.4	51.0	27.0	73.0	76.2	66.5	74.1	89.8	66.3
DCCL (CVPR23)	63.5	60.8	64.9	43.1	55.7	36.2	–	–	–	–	–	–	75.3	76.8	70.2	80.5	90.5	76.2
PromptCAL (CVPR23)	62.9	64.4	62.1	50.2	70.1	40.6	52.2	52.2	52.3	37.0	52.0	28.9	81.2	84.2	75.3	83.1	92.7	78.3
GPC (ICCV23)	55.4	58.2	53.1	42.8	59.2	32.8	46.3	42.5	47.9	–	–	–	77.9	85.0	63.0	76.9	94.3	71.0
SimGCD (ICCV23)	60.3	65.6	57.7	53.8	71.9	45.0	54.2	59.1	51.8	44.0	58.0	36.4	80.1	81.2	77.8	83.0	93.1	77.9
PIM (CVPR23)	62.7	75.7	56.2	43.1	66.9	31.6	–	–	–	42.3	56.1	34.8	78.3	84.2	66.5	83.1	95.3	77.0
TRAILER (CVPR24)	65.1	71.3	61.9	55.4	71.7	47.6	54.5	62.6	50.5	44.5	57.0	37.8	–	–	–	–	–	–
SelEx (ECCV24)	73.6	75.3	72.8	58.5	75.6	50.3	57.1	64.7	53.3	–	–	–	82.3	85.3	76.3	83.1	93.6	77.8
CMS (CVPR24)	68.2	76.5	64.0	56.9	76.1	47.6	56.0	63.4	52.3	36.4	54.9	26.4	82.3	**85.7**	75.5	84.7	**95.6**	79.2
SPT (ICLR24)	65.8	68.8	65.1	59.0	**79.2**	49.3	59.3	61.8	58.1	43.4	58.7	35.2	81.3	84.3	75.6	85.4	93.2	81.4
AGCD (CVPR24)	66.6	66.5	66.7	48.4	57.7	39.3	53.7	51.5	56.0	–	–	–	71.3	75.7	66.8	83.3	90.2	76.5
NC-GCD (NeurIPS25)	74.8	76.8	73.8	59.9	77.8	51.2	60.0	57.6	61.2	46.4	58.4	40.7	82.7	85.5	77.3	88.4	94.1	**85.5**
ProtoGCD (TPAMI25)	65.2	68.5	60.5	53.8	73.7	44.2	56.8	62.5	53.9	44.5	59.4	36.5	81.9	82.9	80.0	84.0	92.2	79.9
InfoSculpt (arXiv26)	66.8	72.0	59.9	59.5	78.4	42.1	57.0	65.0	50.1	48.5	**61.2**	36.0	82.2	83.3	78.0	85.6	93.7	77.5
Ours	**77.0**	**79.1**	**74.9**	**75.7**	75.8	**75.5**	**71.0**	**73.8**	**68.3**	**55.2**	55.4	**55.0**	**84.8**	84.5	**86.1**	**88.7**	93.0	84.5

$T = 0.5$ and retains annotated samples together with samples whose maximum graph probability exceeds 0.65. Graph consistency aligns encoder features for pairs with similar propagated class distributions. For a mini-batch $\mathcal{B}$, let $\mathbf{u}_i = f_\theta(x_i)/\|f_\theta(x_i)\|_2$ and $s_{ij} = \tilde{\mathbf{y}}_i^\top \tilde{\mathbf{y}}_j$, using the propagated distributions before sharpening. Let $q_\mathcal{B}$ be the 80th percentile of all entries in $[s_{ij}]_{i,j\in\mathcal{B}}$, including diagonal entries. The positive-pair set is

$$\mathcal{E}_\mathcal{B}^+ = \{(i,j) \in \mathcal{B} \times \mathcal{B} : i \neq j,\ s_{ij} > \max(q_\mathcal{B}, 0.2)\}\,. \tag{6}$$

We minimize the mean cosine distance over these pairs:

$$\mathcal{L}_{\text{cons}} = \frac{\sum_{(i,j)\in\mathcal{E}_\mathcal{B}^+} \left(1 - \mathbf{u}_i^\top \mathbf{u}_j\right)}{\max\left(1, |\mathcal{E}_\mathcal{B}^+|\right)}. \tag{7}$$

The propagated distributions and selected pairs are held fixed during each model update. The loss is zero when the positive-pair set is empty. The joint-refinement objective is

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{softCE}} + \lambda_{\text{cons}}\mathcal{L}_{\text{cons}}, \tag{8}$$

where $\mathcal{L}_{\text{CE}}$ includes the hard-label loss coefficient, set to two during encoder fine-tuning, and λ_{cons} controls feature consistency. The graph provides targets for the classifier and relations for the encoder. In turn, updated features determine the next neighborhood graph. Inference uses a forward pass through the encoder and adapter to predict the category of each input image.

4 Experiments

4.1 Implementation Details

We evaluate on six datasets. CIFAR-100 [41] and ImageNet100 [42] cover generic objects. CUB-200 [43], Stanford Cars [44], FGVC Aircraft [45], and Herbarium19 [46] cover fine-grained categories. The initial labeled set contains samples from old classes, while the unlabeled training pool contains both old and new classes. PD-GCD then queries labels from this pool under an annotation budget, following the active GCD setting [9]. The Old and New evaluation groups are defined by the initial class split and remain fixed after queries.

The main experiments use a DINO-pretrained ViT-B/16 [47]. The first stage learns representations with supervised and unsupervised GCD objectives. The second stage combines active queries, graph distillation and joint refinement of the adapter and the final encoder block. The final encoder normalization layer remains frozen. Graph construction uses training samples. At evaluation, the encoder and adapter predict categories for test images. We report clustering accuracy over all evaluated images and separately over the Old and New groups. H.M. is $2\,\text{Old}\,\text{New}/(\text{Old}+\text{New})$. The appendix describes the training schedules, query-budget definition, optimizer settings and phase-specific objectives.

Table 2: Macro-average performance over the four fine-grained datasets, two generic datasets, and all six datasets in Table 1. H.M. is computed from the unrounded group means of Old and New accuracy. Improvements are differences between displayed entries.

Method	Fine-grained Avg				Generic Avg				All Avg			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
SimGCD (ICCV23)	53.1	63.7	47.7	54.5	81.6	87.2	77.9	82.2	62.6	71.5	57.8	63.9
CMS (CVPR24)	54.4	67.7	47.6	55.9	83.5	**90.7**	77.4	83.5	64.1	75.4	57.5	65.2
SPT (ICLR24)	56.9	67.1	51.9	58.6	83.4	88.8	78.5	83.3	65.7	74.3	60.8	66.9
NC-GCD (NeurIPS25)	60.3	67.7	56.7	61.7	85.6	89.8	81.4	85.4	68.7	75.0	65.0	69.6
InfoSculpt (arXiv26)	58.0	69.2	47.0	56.0	83.9	88.5	77.8	82.8	66.6	75.6	57.3	65.2
Ours (PD-GCD)	**69.7**	**71.0**	**68.4**	**69.7**	**86.8**	88.8	**85.3**	**87.0**	**75.4**	**76.9**	**74.1**	**75.5**
Improv. over SOTA	**+9.4**	**+1.8**	**+11.7**	**+8.0**	**+1.2**	-1.9	**+3.9**	**+1.6**	**+6.7**	**+1.3**	**+9.1**	**+5.9**

Table 3: Comparison of Anchor Mining and Graph Propagation module configurations on four datasets.

ID	**Configurations**		**CUB**				**Stanford Cars**				**FGVC Aircraft**				**CIFAR100**			
	Anchor Mining	Graph Propagation	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
(a)	✗	✗	60.3	65.6	57.7	61.4	53.8	71.9	45.0	55.4	54.2	59.1	51.8	55.2	80.1	81.2	77.8	79.5
(b)	✓	✗	72.5	76.6	68.4	72.3	69.3	72.8	65.9	69.2	64.4	67.1	61.7	64.3	82.7	83.6	79.3	81.4
(c)	✗	✓	64.1	62.3	65.9	64.0	59.5	58.1	61.0	59.5	64.1	65.9	62.3	64.1	71.5	79.0	41.7	54.6
(d)	✓	✓	**77.0**	**79.1**	**74.9**	**76.9**	**75.7**	**75.8**	**75.5**	**75.6**	**71.0**	**73.8**	**68.3**	**70.9**	**84.8**	**84.5**	**86.1**	**85.3**
Improv. with Anchor Mining			**+12.2**	**+11.0**	**+10.7**	**+10.9**	**+15.5**	**+0.9**	**+20.9**	**+13.8**	**+10.2**	**+8.0**	**+9.9**	**+9.1**	**+2.6**	**+2.4**	**+1.5**	**+1.9**
Improv. with Graph Propagation			**+3.8**	**-3.3**	**+8.2**	**+2.6**	**+5.7**	**-13.8**	**+16.0**	**+4.1**	**+9.9**	**+6.8**	**+10.5**	**+8.9**	**-8.6**	**-2.2**	**-36.1**	**-24.9**
Improv. with Both Mechanisms			**+16.7**	**+13.5**	**+17.2**	**+15.5**	**+21.9**	**+3.9**	**+30.5**	**+20.2**	**+16.8**	**+14.7**	**+16.5**	**+15.7**	**+4.7**	**+3.3**	**+8.3**	**+5.8**

4.2 Comparison with State-of-the-Art Methods

Table 1 compares PD-GCD with GCD and active GCD methods using DINOv1 features on six benchmarks. The comparison includes GPC [48], PIM [49], TRAILER [50], and SPT [51]. PD-GCD acquires additional training labels through active queries. All, Old and New accuracy measure overall recognition and performance on the two initial category groups.

The evaluation covers four fine-grained benchmarks, CUB, Stanford Cars, FGVC Aircraft and Herbarium19, and two generic benchmarks, CIFAR-100 and ImageNet100. Table 2 summarizes the results by dataset group, with H.M. measuring the balance between Old and New accuracy.

4.3 Ablation Study

Table 3 compares four module configurations. Row (a) is SimGCD; row (b) uses Anchor Mining to acquire additional annotations; row (c) applies Graph Propagation to a different base GCD model; and row (d) combines Anchor Mining and Graph Propagation. Rows (b) and (d) use actively queried labels.

The four configurations are compared on CUB, Stanford Cars, FGVC Aircraft and CIFAR-100 using All, Old, New and H.M. These metrics describe recognition across all categories and the balance between the Old and New groups.

Table 4 examines anchor neighborhoods and propagation predictions on Aircraft at a fixed checkpoint. Hit@k records whether a sample has an anchor among its nearest neighbors, AnchorRatio measures the proportion of anchor neighbors, and AnchorPurity measures their semantic consistency. LP Probe accuracy, confidence and entropy characterize the propagated assignments.

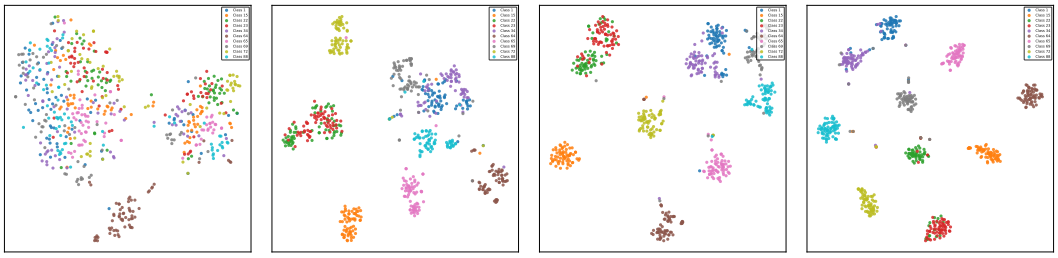
Table 5 compares configurations of three training losses on CUB, Stanford Cars, FGVC Aircraft and CIFAR-100. $\mathcal{L}_{\mathrm{CE}}$ uses the initial and actively acquired labels, $\mathcal{L}_{\mathrm{softCE}}$ transfers graph predictions to the model, and $\mathcal{L}_{\mathrm{cons}}$ relates feature similarity to agreement between graph predictions. The table includes hard-label CE alone, three configurations that omit one loss, and the full objective, evaluated with All, Old, New and H.M.

Table 4: Anchor-graph coupling analysis on FGVC Aircraft. The Label Propagation (LP) Probe reports LabelSpreading performance using propagation anchors as boundary conditions. Lower entropy indicates more confident propagated labels.

Setting	Anchor Count	Anchor-Graph Statistics			LP Probe				
		Hit@k	Anchor Ratio	Anchor Purity	All	Old	New	Conf.	Ent.
After Anchor Mining	2066	**0.9613**	0.2278	0.6197	0.6409	0.6679	0.6138	0.8044	0.4789
After Graph Propagation	2066	0.9433	**0.2296**	**0.7125**	**0.7054**	**0.7326**	**0.6781**	**0.9496**	**0.1232**

Table 5: Ablation of loss components on four benchmarks. We study three losses in our framework: the supervised classification loss $\mathcal{L}_{\text{CE}}$, the graph soft-label distillation loss $\mathcal{L}_{\text{softCE}}$, and the graph consistency loss $\mathcal{L}_{\text{cons}}$. H.M. denotes the harmonic mean of Old and New accuracy.

Setting	Loss Components			CUB				Stanford Cars				FGVC Aircraft				CIFAR100			
	$\mathcal{L}_{\text{CE}}$	$\mathcal{L}_{\text{softCE}}$	$\mathcal{L}_{\text{cons}}$	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
$\mathcal{L}_{\text{CE}}$ only	✓	✗	✗	72.6	74.7	70.4	72.5	77.8	77.6	78.1	77.8	70.2	72.8	67.6	70.1	83.2	82.9	84.4	83.7
w/o $\mathcal{L}_{\text{softCE}}$	✓	✗	✓	72.7	74.6	70.8	72.6	77.9	77.7	78.1	77.9	70.5	72.3	68.7	70.4	83.5	83.5	83.5	83.5
w/o $\mathcal{L}_{\text{cons}}$	✓	✓	✗	74.8	77.2	72.3	74.7	**78.4**	**77.9**	78.8	**78.4**	**71.6**	73.3	**69.9**	**71.5**	83.9	83.6	85.0	84.3
w/o $\mathcal{L}_{\text{CE}}$	✗	✓	✓	74.0	77.7	70.3	73.8	78.2	77.5	**79.0**	78.2	70.5	71.2	69.8	70.5	83.9	83.6	85.4	84.5
Full	✓	✓	✓	**77.0**	**79.1**	**74.9**	**76.9**	75.7	75.8	75.5	75.6	71.0	**73.8**	68.3	70.9	**84.8**	**84.5**	**86.1**	**85.3**



(a) Original (b) CMS[7] (c) SelEx[19] (d) Ours

Figure 5: t-SNE visualization on FGVC Aircraft dataset with 10 randomly selected classes.

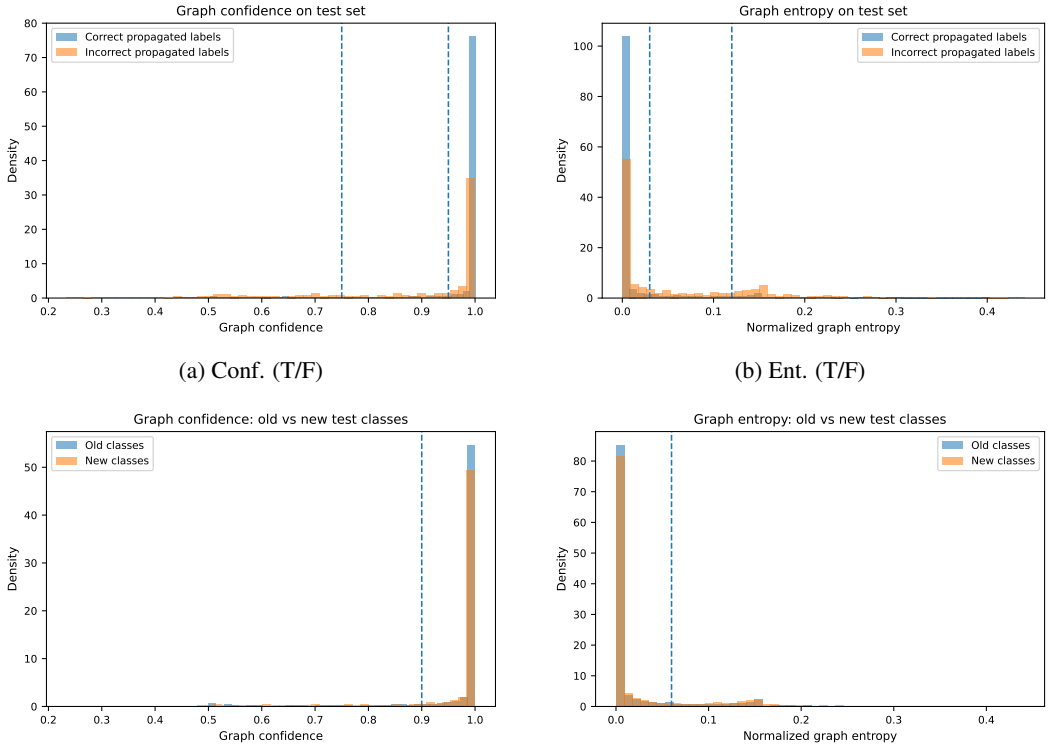
4.4 Visualization Study

Figure 5 compares t-SNE projections for ten selected Aircraft categories. The original representation and the CMS, SelEx and PD-GCD embeddings are visualized to examine within-category compactness and separation between categories.

Figure 6 examines the confidence and entropy of propagated predictions on Aircraft. Predictions are grouped by correctness and by the initial Old/New split, allowing the distributions to be compared across correct and incorrect predictions and across the two category groups.

5 Conclusion

We presented PD-GCD, an active generalized category discovery framework that connects label acquisition, graph propagation, and representation learning. Queried labels seed a neighborhood graph, and the propagated assignments supervise a parametric model through distillation and consistency learning. Graph uncertainty guides the active queries. After acquisition ends, updated features refine the neighborhood structure used for training. Experiments on six benchmarks show strong novel-category recognition, particularly on fine-grained datasets. The mechanism study highlights the role of labeled anchor coverage, and the loss configurations give different rankings across datasets. These findings point to the quality of the interaction between acquired labels and learned neighborhoods as a central factor in active category discovery.



(a) Conf. (T/F)

(b) Ent. (T/F)

(c) Conf. (O/N)

(d) Ent. (O/N)

Figure 6: Graph confidence and entropy analysis on the FGVC Aircraft test set.

Acknowledgments

The authors would like to thank anonymous reviewers for their suggestive comments. This paper is supported by NSFC under grant No. 62495092, No.62125305 and Natural Science Basic Research Plan in Shaanxi Province of China (No. 2025SYS-SYSZD-023).

References

[1] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1893–1902, 2015.

[4] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572, 2016.

[5] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Generalized category discovery. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7492–7501, 2022.

[6] Xin Wen, Bingchen Zhao, and Xiaojuan Qi. Parametric classification for generalized category discovery: A baseline study. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16590–16600, 2023.

[7] Sua Choi, Dahyun Kang, and Minsu Cho. Contrastive mean-shift learning for generalized category discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23094–23104, 2024.

[8] Shijie Ma, Fei Zhu, Xu-Yao Zhang, and Cheng-Lin Liu. Protogcd: Unified and unbiased prototype learning for generalized category discovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[9] Shijie Ma, Fei Zhu, Zhun Zhong, Xu-Yao Zhang, and Cheng-Lin Liu. Active generalized category discovery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16890–16900, 2024.

[10] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.

[11] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16, 2003.

[12] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5070–5079, 2019.

[13] Jizhou Han, Shaokun Wang, Yuhang He, Chenhao Ding, Qiang Wang, Xinyuan Gao, SongLin Dong, and Yihong Gong. Consistent supervised–unsupervised alignment for generalized category discovery. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[14] Enrico Fini, Enver Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. A unified objective for novel class discovery. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9284–9292, 2021.

[15] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European conference on computer vision*, pages 268–285. Springer, 2020.

[16] Zhun Zhong, Enrico Fini, Subhankar Roy, Zhiming Luo, Elisa Ricci, and Nicu Sebe. Neighborhood contrastive learning for novel class discovery. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10862–10870, 2021.

[17] Nan Pu, Zhun Zhong, and Nicu Sebe. Dynamic conceptional contrastive learning for generalized category discovery. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7579–7588, 2023.

[18] Sheng Zhang, Salman Khan, Zhiqiang Shen, Muzammal Naseer, Guangyi Chen, and Fahad Shahbaz Khan. Promptcal: Contrastive affinity learning via auxiliary prompts for generalized novel category discovery. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3479–3488, 2023.

[19] Sarah Rastegar, Mohammadreza Salehi, Yuki M Asano, Hazel Doughty, and Cees GM Snoek. Selex: Self-expertise in fine-grained generalized category discovery. In *European Conference on Computer Vision*, pages 440–458. Springer, 2024.

[20] Bingchen Zhao and Oisin Mac Aodha. Incremental generalized category discovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19137–19147, 2023.

[21] Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

[22] Dan Wang and Yi Shang. A new active labeling method for deep learning. *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 112–119, 2014.

[23] Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pages 413–424. Springer, 2006.

[24] Yarin Gal and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning (ICML)*, 2017.

[25] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 93–102, 2019.

[26] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations (ICLR)*, 2018.

[27] Fedor Zhdanov. Diverse mini-batch active learning. In *arXiv preprint arXiv:1901.05954*, 2019.

[28] Jon Vadillo, Roberto Santana, Jose A. Lozano, and Marta Kwiatkowska. Uncertainty-aware explanations through probabilistic self-explainable neural networks. *arXiv preprint arXiv:2403.13740*, 2025.

[29] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In *International Conference on Learning Representations (ICLR)*, 2020.

[30] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. Variational adversarial active learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5972–5981, 2019.

[31] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005.

[32] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7(11), 2006.

[33] Youqiang Zhang, Guo Cao, Bisheng Wang, Xuesong Li, Prince Yaw Owusu Amoako, and Ayesha Shafique. Dual sparse representation graph-based copropagation for semisupervised hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–17, 2022.

[34] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[35] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

[36] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

[37] 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.

[38] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020.

[39] Chenxiao Yang, Qitian Wu, and Junchi Yan. Geometric knowledge distillation: Topology compression for graph neural networks. *Advances in Neural Information Processing Systems*, 35:29761–29775, 2022.

[40] Dongwon Kim, Namyup Kim, Cuiling Lan, and Suha Kwak. Shatter and gather: Learning referring image segmentation with text supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15569–15581, 2023.

[41] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[42] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[43] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[44] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[45] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[46] Kiat Chuan Tan, Yulong Liu, Barbara Ambrose, Melissa Tulig, and Serge Belongie. The herbarium challenge 2019 dataset. *arXiv preprint arXiv:1906.05372*, 2019.

[47] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[48] Bingchen Zhao, Xin Wen, and Kai Han. Learning semi-supervised gaussian mixture models for generalized category discovery. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16623–16633, 2023.

[49] Florent Chiaroni, Jose Dolz, Ziko Imtiaz Masud, Amar Mitiche, and Ismail Ben Ayed. Parametric information maximization for generalized category discovery. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1729–1739, 2023.

[50] Ruixuan Xiao, Lei Feng, Kai Tang, Junbo Zhao, Yixuan Li, Gang Chen, and Haobo Wang. Targeted representation alignment for open-world semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 23072–23082, 2024.

[51] Hongjun Wang, Sagar Vaze, and Kai Han. Sptnet: An efficient alternative framework for generalized category discovery with spatial prompt tuning. *arXiv preprint arXiv:2403.13684*, 2024.

[52] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-set recognition: A good closed-set classifier is all you need? 2021.

[53] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[54] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[55] Yue Liu, Wenxuan Tu, Sihang Zhou, Xinwang Liu, Linxuan Song, Xihong Yang, and En Zhu. Deep graph clustering via dual correlation reduction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7603–7611, 2022.

[56] Yixin Liu, Yu Zheng, Daokun Zhang, Hongxu Chen, Hao Peng, and Shirui Pan. Towards unsupervised deep graph structure learning. *arXiv e-prints*, 2022.

Appendix

A. More Experimental Settings

A.1. **Dataset Setup** 15

A.2. **Training Parameters and Details** 15

B. Theoretical Motivation 16

C. Overall Algorithm of Our Method

C.1. **Anchor Mining** 17

C.2. **Graph Propagation and Anchor Refinement** 18

D. Additional Experiment Results and Analysis

D.1. **Results with Different Backbones** 19

D.2. **Robustness Analysis under Noisy and Long-Tailed Training Distributions** 19

D.3. **Additional Experiment** 20

E. Additional Hyperparameter Analysis

E.1. **Effect of Graph Consistency Weight** 21

E.2. **Effect of the Query Budget** 22

E.3. **Effect of the GCD Supervision Balance** 23

E.4. **Effect of Pseudo-Label Confidence Threshold** 23

E.5. **Effect of Pseudo-Label Expansion Ratio** 24

E.6. **Effect of Neighborhood Size in Graph Propagation** 24

F. Additional Visualization Analysis 24

G. Limitations and Future Work 25

A More Experimental Settings

A.1 Dataset Setup

The six benchmarks cover generic objects and fine-grained categories. The initial labeled set contains examples from the Old classes. The unlabeled training pool contains further examples from those classes together with examples from the New classes. A queried sample moves from the unlabeled pool to the labeled set. Its Old/New membership remains fixed by the initial class partition.

CUB-200-2011 [43] contains 200 bird species, split into 100 Old and 100 New classes. Stanford Cars [44] contains 196 vehicle categories, split into 98 Old and 98 New classes. FGVC Aircraft [45] contains 100 variants, split into 50 Old and 50 New classes. We use the SSB partitions [52] for these three fine-grained datasets. Herbarium19 [46] contains 683 plant species, with 341 Old and 342 New classes.

CIFAR-100 [41] contains 60,000 images from 100 object classes. We designate 80 classes as Old and 20 as New. ImageNet100 [42] contains 100 selected ImageNet classes, divided into 50 Old and 50 New classes. CIFAR-100 uses class IDs 0–79 as Old and 80–99 as New. The Old/New partition remains fixed throughout training and evaluation. New classes contribute unlabeled training images as well as evaluation images.

A.2 Training Parameters and Details

The main model uses a DINO-pretrained ViT-B/16 encoder. Stage one trains the final Transformer block and a projection head with the GCD objective. Stage two consists of active-query rounds, an adapter-only distillation phase, and joint refinement of the adapter and the final Transformer block. A round begins by rebuilding the graph and generating propagation targets, followed by its assigned training epochs. The encoder stays frozen during querying and adapter-only distillation, and its final normalization layer remains frozen during joint refinement. Encoder features are re-extracted as the representation is updated.

Stage one uses SGD with momentum 0.9, weight decay 10^{-4}, and an initial learning rate of 0.1. Stage two uses Adam with weight decay 10^{-4}. The adapter learning rate is 10^{-3}, and the trainable encoder block uses 2×10^{-5} during joint refinement. Learning rates remain constant within the second stage. Adapter training from cached features uses batches of 256, and joint refinement uses image batches of 32. The adapter has a 2048-dimensional hidden layer, dropout 0.1, and a 512-dimensional bottleneck followed by the category classifier. The graph uses $k = 20$ neighbors, similarity temperature 0.07, and propagation coefficient $\alpha = 0.2$, with 30 and 50 iterations for the two propagation passes. The default first-stage supervision balance is $\lambda_{\text{sup}} = 0.5$.

Queries acquire ground-truth labels for selected training samples. Let B_1 and B_2 denote the numbers of labels acquired in stages one and two, respectively. The total additional annotation budget is $B_1 + B_2$. Stage two allocates its planned budget across the active rounds, with each query limited by the remaining unlabeled pool. The query strategy ranks candidates by graph entropy, forms a pool of up to five times the query size, and selects diverse representatives by clustering. Self-training uses the resulting annotated set and graph predictions. For equal per-stage budgets $B_1 = B_2 = B$, the total is $2B$.

Images are resized to 224×224 and augmented with random crops and horizontal flips. Stage-one contrastive learning uses two views per image and a sampler that balances labeled and unlabeled examples. Graph seed expansion retains at most the most confident 10% of candidates within each predicted category, subject to a confidence threshold of 0.85. During active querying, adapter training uses hard-label cross-entropy. During adapter-only distillation, hard-label cross-entropy and soft-label cross-entropy have unit weight, and distillation uses all training samples at temperature $T = 1$. During joint refinement, the hard-label coefficient is 2, the distillation coefficient is 1, and the consistency coefficient is 0.5. Distillation in this phase uses temperature $T = 0.5$ and includes annotated samples and predictions with maximum graph probability above 0.65. Consistency uses the positive pairs in Eq. (6) and the mean cosine distance in Eq. (7). Pair selection uses propagated distributions before sharpening, with the 80th percentile computed over the full batch similarity matrix and self-pairs excluded from the loss.

Round and epoch schedules. Let R_a and R_s be the numbers of active-query and self-training rounds, and let F be the number of initial self-training rounds with a frozen encoder. Active rounds, frozen distillation rounds and joint-refinement rounds contain E_a, E_f and E_j epochs, respectively. The second-stage epoch count is $R_a E_a + F E_f + (R_s - F) E_j$. The 50-epoch configuration uses $(R_a, R_s, F) = (10, 40, 5)$ and $E_a = E_f = E_j = 1$.

B Theoretical Motivation

An idealized graph model helps explain how queried labels affect other samples. Consider an undirected graph with symmetric nonnegative affinities. Every connected component containing unseeded nodes must have a labeled seed. These conditions make the harmonic extension below well defined. In active GCD, queries can supply seeds from both Old and New classes, so propagation can assign semantic information beyond the classes labeled at initialization.

At iteration t, PD-GCD embeds samples from the current labeled set and the remaining unlabeled pool as

$$\mathbf{v}_i = \mathrm{normalize}(b_\psi(f_\theta(x_i))), \qquad x_i \in \mathcal{D}_l \cup \mathcal{D}_u.$$

Here b_ψ denotes the adapter bottleneck. We construct a k-nearest-neighbor graph in this representation space. Let $\mathbf{W}$ denote the affinity matrix, where $\mathbf{W}_{ij} \geq 0$ measures the local similarity between samples x_i and x_j. Let $\mathbf{D}_g$ be the degree matrix with $(\mathbf{D}_g)_{ii} = \sum_j \mathbf{W}_{ij}$, and define the graph Laplacian as

$$\mathbf{L}_g = \mathbf{D}_g - \mathbf{W}.$$

The graph Laplacian characterizes the smoothness of semantic assignments over the data manifold: assigning very different soft labels to strongly connected samples leads to a large graph smoothness energy.

Anchor Mining identifies a propagation anchor set $A^t \subset \mathcal{D}_u$ from the unlabeled pool, where

$$A^t = A^t_{\mathrm{cen}} \cup A^t_{\mathrm{bd}}.$$

The oracle labels the selected anchors, which then join the annotated set. Initial annotations and acquired anchor labels provide one-hot seed assignments. The remaining samples receive predictions from graph propagation. High-confidence pseudo-label expansion can add provisional seeds in a second pass, but those predictions remain distinct from acquired labels.

Let $\tilde{\mathbf{Y}} = [\tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_N]^\top$ denote the propagated soft-label matrix. From the harmonic-extension perspective, Graph Propagation estimates $\tilde{\mathbf{Y}}$ by minimizing the graph smoothness energy:

$$\min_{\tilde{\mathbf{Y}}} \frac{1}{2} \sum_{i,j} \mathbf{W}_{ij} \|\tilde{\mathbf{y}}_i - \tilde{\mathbf{y}}_j\|_2^2.$$

Equivalently, this objective can be written as

$$\min_{\tilde{\mathbf{Y}}} \mathrm{Tr}\left(\tilde{\mathbf{Y}}^\top \mathbf{L}_g \tilde{\mathbf{Y}}\right).$$

Initial and queried labels constrain the seed nodes. The objective extends those assignments to the remaining graph while penalizing disagreement between strongly connected neighbors. Our implementation uses iterative weighted propagation with clamped seeds. The symmetric graph formulation isolates the role of seed placement and provides an interpretation of this procedure.

This formulation also clarifies the role of Anchor Mining. Following the standard graph-propagation notation, we partition the graph Laplacian into seed and remaining-node blocks:

$$\mathbf{L}_g = \begin{bmatrix} \mathbf{L}_{ss} & \mathbf{L}_{sr} \\ \mathbf{L}_{rs} & \mathbf{L}_{rr} \end{bmatrix}.$$

The seed block contains the currently annotated nodes. $\mathbf{Y}_s$ stacks their one-hot labels, and $\tilde{\mathbf{Y}}_r$ denotes assignments for the remaining nodes. Under the connectivity conditions above, $\mathbf{L}_{rr}$ is invertible and the harmonic solution satisfies

$$\mathbf{L}_{rr} \tilde{\mathbf{Y}}_r + \mathbf{L}_{rs} \mathbf{Y}_s = 0,$$

and hence

$$\tilde{\mathbf{Y}}_r^\star = -\mathbf{L}_{rr}^{-1}\mathbf{L}_{rs}\mathbf{Y}_s.$$

The solution depends on both the graph and the seed assignments. Representative anchors place labels near cluster centers. The additional samples with large nearest-center distance extend the queried set to less represented regions. These two criteria target different aspects of graph coverage.

The same dependence can be seen from a perturbation perspective. If the seed assignments are perturbed by Δ_s, the induced perturbation on the propagated labels satisfies

$$\Delta_r = -\mathbf{L}_{rr}^{-1}\mathbf{L}_{rs}\Delta_s,$$

and therefore

$$\|\Delta_r\|_2 \le \left\|\mathbf{L}_{rr}^{-1}\mathbf{L}_{rs}\right\|_2 \|\Delta_s\|_2.$$

The bound shows how errors in seed assignments can affect unseeded nodes through the graph. This matters especially for pseudo-label expansion, where a confident prediction can still be wrong. Querying provides true labels for selected nodes, while confidence filtering limits which predictions become provisional seeds. The propagation operator determines how either type of seed influences the rest of the graph. This motivates studying seed placement and neighborhood quality together.

C Overall Algorithm of Our Method

Algorithm 1 describes the operations within a stage-one query round. Representation learning and label acquisition repeat until the stage budget is spent. K-means selects representative candidates, and distance-based coverage fills the remaining query slots. The oracle labels the selected samples before they enter subsequent supervised training. The final acquired set supplies additional seeds for stage-two propagation.

C.1 Anchor Mining

Algorithm 1 GCD-Driven Representation Learning with Anchor Mining

Require: Current labeled set $\mathcal{D}_l = \{(x_i, y_i)\}_{i=1}^{N_l}$, remaining unlabeled pool $\mathcal{D}_u = \{x_j\}_{j=1}^{N_u}$, query size M, supervision weight λ_{sup}

Ensure: Updated encoder f_θ, projector g_ϕ, and annotated anchors A^t

1: Reuse the current encoder f_θ and projector g_ϕ, initialized before the first query round.

2: Train (f_θ, g_ϕ) on $\mathcal{D}_l \cup \mathcal{D}_u$ by minimizing

$$\mathcal{L}_{\text{GCD}} = \lambda_{\text{sup}}\mathcal{L}_{\text{sup}} + (1-\lambda_{\text{sup}})\mathcal{L}_{\text{unsup}},$$

where

$$\mathcal{L}_{\text{sup}} = \mathcal{L}_{\text{supCE}} + \mathcal{L}_{\text{supCon}}, \qquad \mathcal{L}_{\text{unsup}} = \mathcal{L}_{\text{cluster}} + \mathcal{L}_{\text{InfoNCE}}.$$

3: Extract features for unlabeled samples:

$$\mathbf{f}_i = f_\theta(x_i), \qquad x_i \in \mathcal{D}_u.$$

4: ℓ_2-normalize $\{\mathbf{f}_i\}$ and perform K-means clustering in the feature space.

5: Mine centroid anchors by selecting samples closest to cluster centers, forming A_{cen}^t.

6: Fill remaining query slots with samples far from their nearest cluster centers, forming A_{bd}^t.

7: Construct the anchor set

$$A^t = A_{\text{cen}}^t \cup A_{\text{bd}}^t, \qquad |A^t| = M.$$

8: Query oracle labels for A^t, add these samples to the labeled set, and remove them from the unlabeled pool.

9: **return** f_θ, g_ϕ, A^t

Table 6: Comparison with state-of-the-art GCD methods using the DINOv2 backbone.

Method	CUB			Stanford Cars			FGVC Aircraft			CIFAR100			ImageNet100		
	All	Old	New	All	Old	New	All	Old	New	All	Old	New	All	Old	New
GCD (CVPR22)	71.9	71.2	72.3	55.4	47.9	59.7	49.7	47.9	50.5	79.6	84.5	69.9	78.5	89.5	73.0
μGCD (NIPS23)	74.6	75.0	74.4	60.6	64.7	58.6	60.0	62.0	59.0	–	–	–	–	–	–
CiPR (CVPR23)	78.3	73.4	80.8	66.7	77.0	61.8	–	–	–	**90.3**	89.0	**93.1**	88.2	87.6	**88.5**
SPT (ICLR24)	76.3	79.5	74.6	–	–	–	–	–	–	–	–	–	90.1	**96.1**	87.1
SimGCD (ICCV23)	71.5	78.1	68.3	71.5	81.9	66.6	63.9	69.9	60.9	88.5	89.2	87.2	89.9	95.5	87.1
SelEx (ECCV24)	87.4	85.1	88.5	79.8	82.3	78.6	82.2	**93.7**	76.7	–	–	–	–	–	–
RLCD (ICML25)	78.7	79.5	78.3	79.5	**91.8**	73.5	72.6	73.3	70.3	–	–	–	–	–	–
ProtoGCD (TPAMI25)	75.7	81.5	72.9	77.6	90.5	71.5	71.1	76.3	68.5	–	–	–	–	–	–
NC-GCD (NeurIPS25)	87.9	85.3	**89.2**	81.1	90.9	79.3	**83.1**	88.3	**82.5**	–	–	–	–	–	–
Ours (PD-GCD)	**88.4**	**89.7**	87.2	**90.6**	90.7	**90.5**	81.5	82.1	81.0	89.3	**89.6**	88.4	**92.1**	**96.1**	88.2

C.2 Graph Propagation and Anchor Refinement

Algorithm 2 Graph Propagation and Anchor Refinement

Require: Labeled set $\mathcal{D}_l$, unlabeled pool $\mathcal{D}_u$, encoder f_θ, adapter a_ψ, budget B_2, graph degree k, expansion threshold τ, consistency weight λ_{cons}, schedule $(R_a, R_s, F, E_a, E_f, E_j)$
Ensure: Refined encoder, adapter and annotated set
Freeze the encoder; set the planned query size to B_2/R_a, capped by the remaining pool.
for $t = 1, \ldots, R_a + R_s$ **do**
  Compute normalized adapter features for $\mathcal{D}_l \cup \mathcal{D}_u$; refresh encoder features during joint refinement.
  Build a k-nearest-neighbor graph and clamp annotated nodes to their one-hot labels.
  Propagate labels, expand seeds above threshold τ, and propagate again to obtain $\tilde{\mathbf{y}}$.
  if $t \leq R_a$ **then**
    Train the adapter for E_a epochs using hard-label cross-entropy on $\mathcal{D}_l$.
    Rank $\mathcal{D}_u$ by graph entropy and select diverse representatives from the high-entropy pool.
    Query their labels and move these samples from $\mathcal{D}_u$ to $\mathcal{D}_l$.
  else if $t \leq R_a + F$ **then**
    Keep the encoder frozen and train the adapter for E_f epochs with hard-label and soft-label cross-entropy, using $T = 1$.
  else
    if $t = R_a + F + 1$ **then**
      Unfreeze the final Transformer block; keep the final encoder normalization layer frozen.
      Initialize the joint optimizer for the trainable block and the adapter.
    end if
    Sharpen graph targets at $T = 0.5$ and retain annotated samples and predictions with confidence above 0.65 for distillation.
    Select positive pairs from $\tilde{\mathbf{y}}$ using Eq. (6) and compute $\mathcal{L}_{\text{cons}}$ using Eq. (7).
    Jointly train for E_j epochs with hard-label coefficient 2, distillation coefficient 1, and consistency coefficient λ_{cons}.
  end if
end for
return f_θ, a_ψ, $\mathcal{D}_l$

Algorithm 2 describes the three training phases. Active rounds alternate adapter training with label acquisition. Frozen self-training rounds distill graph targets into the adapter. Joint refinement updates the adapter and the last Transformer block. Each round rebuilds the graph from the current representation. Acquired labels provide hard supervision and propagation seeds, while graph predictions provide distillation targets and pairwise consistency signals.

Table 7: Comparison with state-of-the-art GCD methods using the CLIP backbone.

Method	CUB			Stanford Cars			FGVC Aircraft			CIFAR100			ImageNet100		
	All	Old	New	All	Old	New	All	Old	New	All	Old	New	All	Old	New
GCD (CVPR22)	51.1	56.2	48.6	62.5	73.9	57.0	41.2	43.0	40.2	70.4	79.3	52.6	71.6	86.0	64.6
PromptCAL (CVPR23)	53.7	61.4	49.9	60.1	77.9	51.5	42.2	48.4	39.0	69.4	77.3	53.5	75.2	87.0	69.3
CMS (CVPR24)	65.8	75.3	61.1	77.9	89.0	72.6	50.3	59.1	45.9	80.3	**85.2**	76.0	85.9	93.8	82.0
NC-GCD (NeurIPS25)	78.5	79.3	78.0	79.0	**90.6**	73.4	63.8	57.6	**63.8**	–	–	–	–	–	–
Ours (PD-GCD)	**81.4**	**83.3**	**79.5**	**85.2**	84.0	**86.4**	**64.6**	**65.6**	63.6	**81.9**	81.3	**84.3**	**89.5**	**94.4**	**84.8**

Table 8: Comparative analysis with multiple representative methods under DINOv2 and CLIP backbones on fine-grained and generic datasets. Fine-grained averages cover CUB, Stanford Cars, and FGVC Aircraft; generic averages cover CIFAR100 and ImageNet100. Each average uses results from all datasets in its group. H.M. denotes the harmonic mean of Old and New accuracy.

	Method	Fine-grained Avg				Generic Avg				All Avg			
		All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
DINOv2	GCD (CVPR22)	59.0	55.7	60.8	58.1	79.1	87.0	71.5	78.5	67.0	68.2	65.1	66.6
	μGCD (NIPS23)	65.1	67.2	64.0	65.6	–	–	–	–	–	–	–	–
	CiPR (CVPR23)	–	–	–	–	89.3	88.3	**90.8**	89.5	–	–	–	–
	SimGCD (ICCV23)	69.0	76.6	65.3	70.5	89.2	92.4	87.2	89.7	77.1	82.9	74.0	78.2
	SelEx (ECCV24)	83.1	87.0	81.3	84.1	–	–	–	–	–	–	–	–
	RLCD (ICML25)	76.9	81.5	74.0	77.6	–	–	–	–	–	–	–	–
	ProtoGCD (TPAMI25)	74.8	82.8	71.0	76.4	–	–	–	–	–	–	–	–
	NC-GCD (NeurIPS25)	84.0	**88.2**	83.7	85.9	–	–	–	–	–	–	–	–
	Ours (PD-GCD)	**86.8**	87.5	**86.2**	**86.9**	**90.7**	**92.9**	88.3	**90.5**	**88.4**	**89.6**	**87.1**	**88.3**
	Improv. over SOTA	**+2.8**	**-0.7**	**+2.5**	**+1.0**	**+1.4**	**+0.5**	**-2.5**	**+0.8**	**+11.3**	**+6.7**	**+13.1**	**+10.1**
CLIP	GCD (CVPR22)	51.6	57.7	48.6	52.8	71.0	82.7	58.6	68.6	59.4	67.7	52.6	59.2
	PromptCAL (CVPR23)	52.0	62.6	46.8	53.6	72.3	82.2	61.4	70.3	60.1	70.4	52.6	60.2
	CMS (CVPR24)	64.7	74.5	59.9	66.4	83.1	**89.5**	79.0	83.9	72.0	80.5	67.5	73.4
	NC-GCD (NeurIPS25)	73.8	75.8	71.7	73.7	–	–	–	–	–	–	–	–
	Ours (PD-GCD)	**77.1**	**77.6**	**76.5**	**77.0**	**85.7**	87.9	**84.6**	**86.2**	**80.5**	**81.7**	**79.7**	**80.7**
	Improv. over SOTA	**+3.3**	**+1.8**	**+4.8**	**+3.3**	**+2.6**	**-1.6**	**+5.6**	**+2.3**	**+8.5**	**+1.2**	**+12.2**	**+7.3**

D Additional Experiment Results and Analysis

D.1 Results with Different Backbones

Tables 6 and 7 compare recognition results using DINOv2 [53] and CLIP [54] backbones on CUB, Stanford Cars, FGVC Aircraft, CIFAR100 and ImageNet100. Each table reports All, Old and New accuracy. Table 8 summarizes the fine-grained, generic and overall dataset averages, together with H.M., to examine recognition across backbone choices and category groups.

D.2 Robustness Analysis under Noisy and Long-Tailed Training Distributions

Table 9 examines perturbations to the graph representation. Zero-mean Gaussian noise is added to adapted training features before graph construction, and the perturbed features are normalized again before neighbor search. The noise standard deviations are 0.1, 0.3 and 0.5, with zero noise as the baseline. All, Old, New and H.M. characterize recognition under these perturbations.

For the long-tailed setting, we follow a class-wise long-tail subsampling protocol to construct imbalanced training data. Specifically, for each training split, including both labeled and unlabeled subsets (i.e., `imbalance_target=both`), class i is retained according to an exponential decay factor

$$r_i = \mathrm{IR}^{-i/(C-1)},$$

where C is the number of classes present in the corresponding training subset and $\mathrm{IR} \in \{10, 50\}$. Classes are ordered by their original identifiers before label remapping.

The retained number of samples for each class is then determined by

$$n_i' = \max(1, \mathrm{round}(n_i r_i)),$$

Table 9: PD-GCD under Gaussian perturbations to adapted training features before graph construction. Noise level denotes standard deviation.

Noise Level	CUB				FGVC Aircraft				CIFAR100				ImageNet100			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
Baseline (Noise 0)	**77.0**	**79.1**	**74.9**	**76.9**	**71.0**	**73.8**	**68.3**	**70.9**	**84.8**	84.5	**86.1**	**85.3**	**88.7**	93.0	**84.5**	**88.5**
Noise 0.1	73.2	76.8	69.6	73.0	68.6	71.1	66.2	68.6	81.0	84.7	66.0	74.2	85.1	90.4	79.8	84.8
Noise 0.3	74.1	77.0	71.2	74.0	67.9	70.4	65.3	67.8	81.5	84.8	68.1	75.5	84.7	92.8	76.6	83.9
Noise 0.5	73.6	76.2	71.1	73.6	66.4	67.9	64.8	66.3	81.1	**85.1**	65.4	73.9	84.9	**93.2**	76.6	84.1

Table 10: Robustness of PD-GCD under long-tailed training distributions with different imbalance ratios across CUB, FGVC Aircraft, CIFAR100, and ImageNet100.

Imbalance Ratio	CUB				FGVC Aircraft				CIFAR100				ImageNet100			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
Original distribution	**77.0**	**79.1**	**74.9**	**76.9**	**71.0**	**73.8**	**68.3**	**70.9**	**84.8**	**84.5**	**86.1**	**85.3**	**88.7**	**93.0**	**84.5**	**88.5**
IR=10	71.0	73.6	68.4	70.9	67.8	71.7	63.9	67.6	81.6	83.9	72.6	77.8	85.4	91.8	79.1	85.0
IR=50	62.9	65.3	60.5	62.8	61.9	66.6	57.2	61.5	79.4	83.4	63.8	72.2	86.9	92.6	81.2	86.5

followed by sampling without replacement. The procedure is applied to both the labeled and unlabeled training subsets. The evaluation set is unchanged, and the baseline row denotes the original distribution.

Table 10 compares recognition on the original training distribution and the long-tailed distributions with $\mathrm{IR} = 10$ and $\mathrm{IR} = 50$. All, Old, New and H.M. describe performance under class imbalance. Old/New groups follow the initial class partition, while head/tail groups describe sample frequency.

D.3 Additional Experiment

Table 11 compares PD-GCD with DCRN and SUBLIME on CUB, FGVC Aircraft, CIFAR100 and ImageNet100. The comparison reports All, Old and New accuracy to examine overall recognition and performance within each category group.

Table 12 compares Graph Diffusion, Heat Diffusion and PD-GCD on CUB, FGVC Aircraft, CIFAR100 and ImageNet100. The experiment examines recognition with different graph-based soft-supervision strategies using All, Old, New and H.M.

PD-GCD stores the neighborhood graph sparsely, with storage proportional to the number of samples times the neighborhood size. Graph construction and propagation run at each graph refresh. Exact nearest-neighbor search has a quadratic pair-search cost.

Tables 13 and 14 examine computational cost on one RTX 4090. Table 13 reports backbone parameter counts and FLOPs, peak memory, training time and inference throughput. Table 14 measures graph construction and propagation time, peak graph memory and overhead relative to the baseline epoch time. The graph timing workloads contain approximately 10,000 Aircraft samples, 11,800 CUB samples and 60,000 CIFAR-100 samples; training-graph sizes follow the training splits.

For the 60,000-sample benchmark, a dense float32 affinity matrix would require 14.4 GB, or about 13.4 GiB, for its entries alone. The measured graph procedure peaks at 1.08 GB. This difference makes sparse graph storage useful at the evaluated scale, while the 37.5-second refresh cost remains relevant when choosing how often to rebuild the graph.

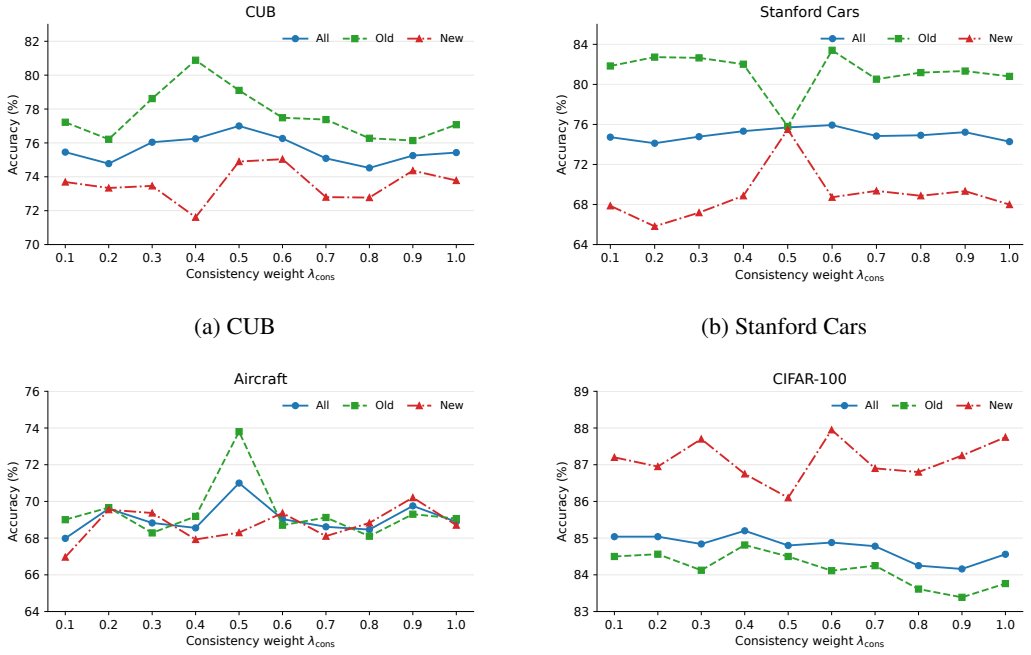
Table 15 compares random, centroid and boundary candidate selection on FGVC Aircraft. Features are partitioned into 100 clusters, and each strategy selects five samples per cluster, yielding 500 candidates. Random selection samples within each cluster; centroid selection favors the smallest nearest-center distance; boundary selection favors the smallest difference between distances to the two nearest centers. Clustering labels are aligned with ground-truth classes by Hungarian matching. Prediction accuracy and class coverage characterize candidate quality, while nearest-center distance and boundary margin describe their locations in feature space.

Table 11: Reported performance of graph-based contrastive methods and PD-GCD. PD-GCD uses actively acquired labels.

Method	CUB			FGVC Aircraft			CIFAR100			ImageNet100		
	All	Old	New	All	Old	New	All	Old	New	All	Old	New
DCRN [55]	69.3	75.1	63.5	63.3	68.6	57.9	79.1	81.3	70.4	81.4	90.2	72.6
SUBLIME [56]	56.1	70.4	41.8	49.2	58.1	40.4	77.8	80.2	68.2	77.6	87.7	67.4
Ours	**77.0**	**79.1**	**74.9**	**71.0**	**73.8**	**68.3**	**84.8**	**84.5**	**86.1**	**88.7**	**93.0**	**84.5**

Table 12: Comparison with diffusion-based soft-supervision strategies across CUB, FGVC Aircraft, CIFAR100, and ImageNet100.

Method	CUB				FGVC Aircraft				CIFAR100				ImageNet100			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
Graph Diffusion	**77.1**	**79.7**	74.6	**77.1**	70.5	72.5	**68.5**	70.4	81.7	84.4	70.9	77.1	87.0	92.3	81.6	86.7
Heat Diffusion	76.6	79.6	73.6	76.5	69.3	70.7	67.9	69.3	82.9	84.2	78.1	81.0	88.3	92.2	**84.5**	88.2
Ours	77.0	79.1	**74.9**	76.9	**71.0**	**73.8**	68.3	**70.9**	**84.8**	**84.5**	**86.1**	**85.3**	**88.7**	**93.0**	**84.5**	**88.5**



(a) CUB

(b) Stanford Cars

(c) FGVC Aircraft

(d) CIFAR100

Figure 7: All, Old and New scores at graph consistency weights λ_{cons} from 0.1 to 1.0 on CUB, Stanford Cars, FGVC Aircraft, and CIFAR100.

E Additional Hyperparameter Analysis

E.1 Effect of Graph Consistency Weight

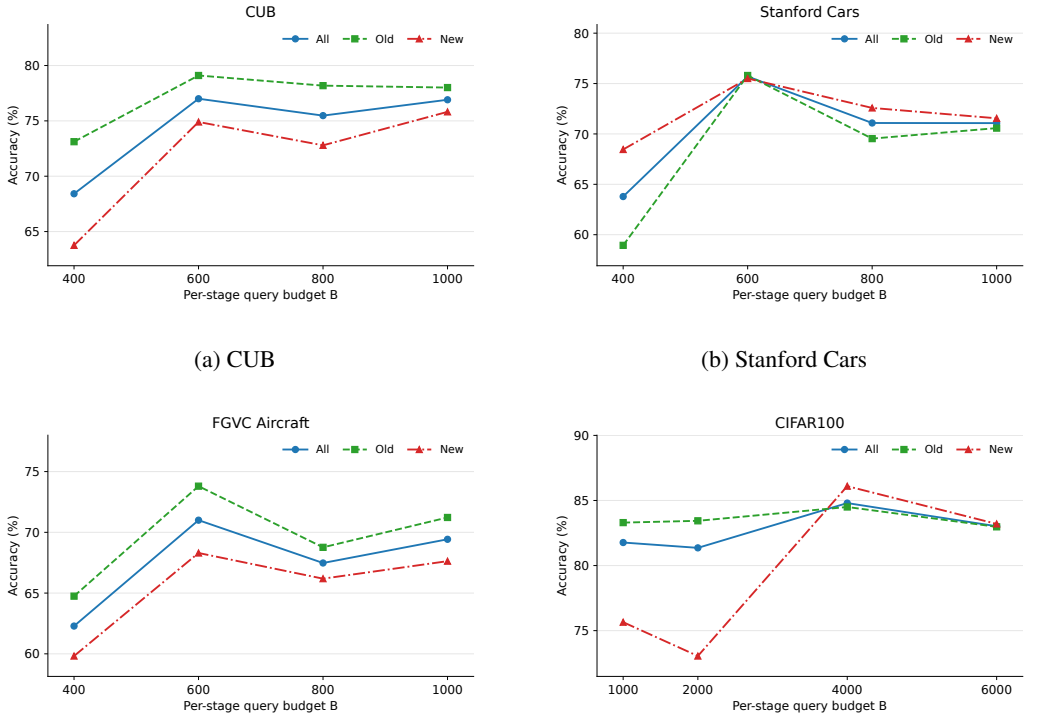
The consistency weight λ_{cons} controls the contribution of graph consistency to feature refinement. Figure 7 presents All, Old and New scores for weights from 0.1 to 1.0 on CUB, Stanford Cars, FGVC Aircraft and CIFAR100. The three metrics describe overall recognition and performance on the initial Old and New category groups.

Table 13: Efficiency and computational overhead on a single RTX 4090.

Dataset	Method	Params	FLOPs	Peak Mem.	Train Time	FPS
FGVC Aircraft (∼10,000 samples)	SimGCD (Baseline)	85.8M	33.7G	∼2.3GB	112.0	∼691
	PD-GCD (Ours)	85.8M	33.7G	∼2.3GB	116.5	∼691
CUB (∼11,800 samples)	SimGCD (Baseline)	85.8M	33.7G	∼2.3GB	135.0	∼172
	PD-GCD (Ours)	85.8M	33.7G	∼2.3GB	141.8	∼172
CIFAR100 (60,000 samples)	SimGCD (Baseline)	85.8M	33.7G	∼2.9GB	420.0	∼190
	PD-GCD (Ours)	85.8M	33.7G	∼2.9GB	457.5	∼190

Table 14: Computational overhead of graph construction and propagation on a single RTX 4090.

Dataset	Samples (N)	Peak Memory	Graph & Prop. Time	Baseline Epoch Time	Overhead
FGVC Aircraft	∼10,000	0.53 GB	4.5s	112s	+4.0%
CUB	∼11,800	0.55 GB	6.8s	135s	+5.0%
CIFAR100	60,000	1.08 GB	37.5s	420s	+8.9%



(a) CUB

(b) Stanford Cars

(c) FGVC Aircraft

(d) CIFAR100

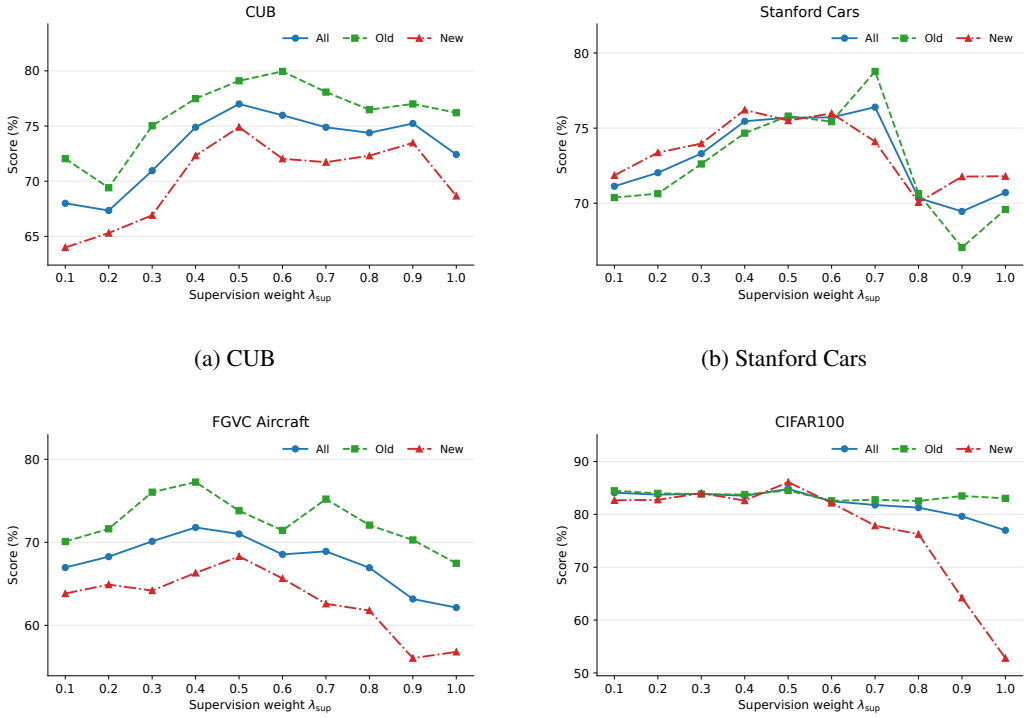
Figure 8: Query-budget results on four datasets. The horizontal axis denotes the per-stage query budget B.

E.2 Effect of the Query Budget

Figure 8 presents All, Old and New scores for different per-stage query budgets on CUB, Stanford Cars, FGVC Aircraft and CIFAR100. The horizontal axis specifies the number of additional labels acquired per stage, allowing recognition to be examined across annotation budgets.

Table 15: Candidate quality analysis on FGVC Aircraft. AnchorAcc is prediction accuracy after clustering-label alignment. ClassCoverage and CoverageRatio denote the number and percentage of ground-truth classes covered. CenterDist is the mean nearest-center distance, and BoundaryMargin is the mean difference between distances to the two nearest centers.

Candidate Type	Candidate Count	Candidate Quality			Candidate Geometry	
		AnchorAcc (%)	Class Coverage	Coverage Ratio (%)	Center Dist	Boundary Margin
Random Candidates	500	72.0	**100**	**100.0**	0.431	0.475
Centroid Candidates	500	**90.0**	99	99.0	0.265	0.654
Boundary Candidates	500	48.2	98	98.0	0.617	0.136



(a) CUB

(b) Stanford Cars

(c) FGVC Aircraft

(d) CIFAR100

Figure 9: All, Old and New scores across supervision weights on four datasets.

E.3 Effect of the GCD Supervision Balance

The supervision weight λ_{sup} balances supervised classification and supervised contrastive learning against clustering and instance contrast. The supervised terms use both initial annotations and labels acquired through queries, including labels from initially Old and initially New classes.

Figure 9 presents All, Old and New scores for supervision weights from 0.1 to 1.0 on CUB, Stanford Cars, FGVC Aircraft and CIFAR100. This comparison examines the balance between supervised and unsupervised representation learning.

E.4 Effect of Pseudo-Label Confidence Threshold

The confidence threshold determines which first-pass predictions may become provisional seeds in the second propagation pass. Table 16 compares thresholds of 0.8, 0.85 and 0.9 using All, Old, New and H.M.

Table 16: Impact of pseudo-label confidence threshold on performance across CUB, FGVC Aircraft, CIFAR100, and ImageNet100.

Pseudo-Label Confidence	CUB				FGVC Aircraft				CIFAR100				ImageNet100			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
0.8	76.4	78.5	74.2	76.3	69.2	71.9	66.5	69.1	84.6	84.4	85.3	84.9	87.2	91.8	82.6	87.0
0.85 (Ours)	**77.0**	79.1	**74.9**	**76.9**	**71.0**	**73.8**	**68.3**	**70.9**	**84.8**	**84.5**	**86.1**	**85.3**	**88.7**	**93.0**	**84.5**	**88.5**
0.9	76.2	**81.0**	71.3	75.8	69.3	70.8	67.7	69.2	84.6	84.4	85.3	84.9	87.4	91.4	83.4	87.2

Table 17: Impact of pseudo-label expansion ratio on performance across CUB, FGVC Aircraft, CIFAR100, and ImageNet100.

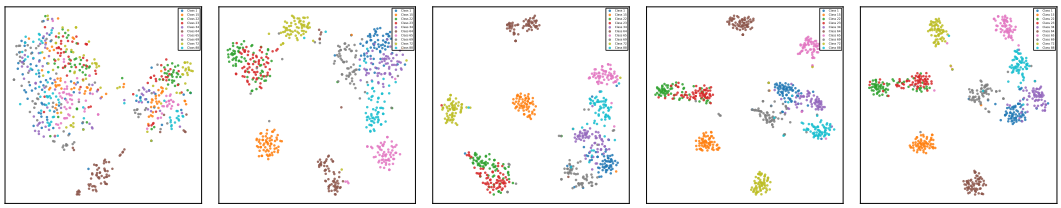
Expansion Ratio	CUB				FGVC Aircraft				CIFAR100				ImageNet100			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
0.05	76.1	81.1	71.2	75.8	68.5	70.3	66.6	68.4	**85.0**	**84.6**	86.5	**85.6**	87.0	91.4	82.7	86.8
0.1 (Ours)	**77.0**	79.1	**74.9**	**76.9**	**71.0**	**73.8**	**68.3**	**70.9**	84.8	84.5	86.1	85.3	**88.7**	**93.0**	**84.5**	**88.5**
0.2	76.4	**81.2**	71.7	76.1	69.1	70.7	67.5	69.1	84.9	84.5	**86.7**	**85.6**	86.9	91.8	82.0	86.6

E.5 Effect of Pseudo-Label Expansion Ratio

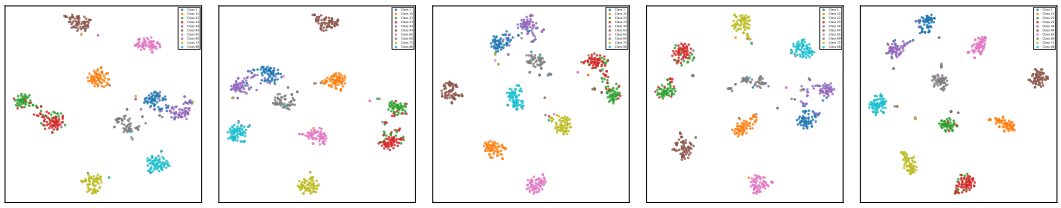
The expansion ratio sets the maximum fraction of candidates retained within each predicted category. Table 17 compares ratios of 0.05, 0.1 and 0.2 using All, Old, New and H.M.

E.6 Effect of Neighborhood Size in Graph Propagation

The neighborhood size controls how many local relations enter graph propagation. Table 18 compares $k = 10$, 20 and 30 using All, Old, New and H.M.



(a) epoch 0 (b) epoch 40 (c) epoch 80 (d) epoch 120 (e) epoch 150



(f) epoch 160 (g) epoch 170 (h) epoch 180 (i) epoch 190 (j) epoch 200

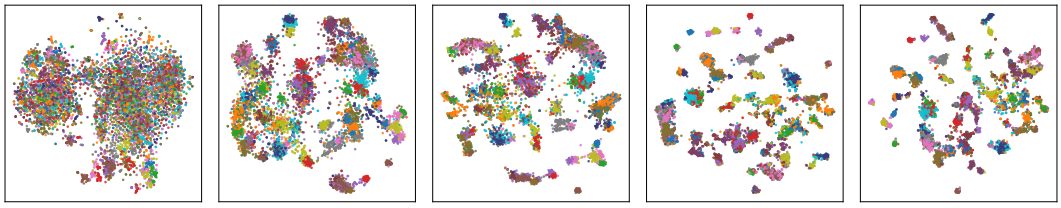
Figure 10: t-SNE visualization of our training process on FGVC Aircraft with 10 randomly selected categories. The first five subfigures show the GCD-driven representation learning phase, while the remaining five show the Graph Propagation refinement phase.

F Additional Visualization Analysis

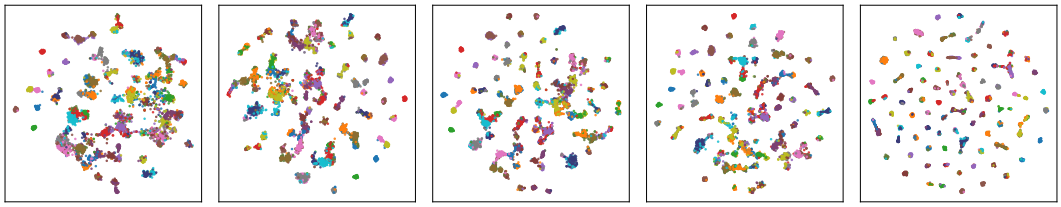
Figures 10 and 11 visualize the evolution of PD-GCD features on FGVC Aircraft using t-SNE for ten selected categories and all categories, respectively. Epochs are counted cumulatively across the two training stages: the first 150 epochs cover representation learning, followed by 50 epochs of graph propagation and model refinement, with epoch 200 marking the end of training. As training progresses, several categories form more compact clusters with clearer separation, while some overlapping regions remain in the all-category view.

Table 18: Impact of neighborhood size k on performance across CUB, FGVC Aircraft, CIFAR100, and ImageNet100.

Neighborhood Size k	CUB				FGVC Aircraft				CIFAR100				ImageNet100			
	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.	All	Old	New	H.M.
10	76.8	**81.2**	72.5	76.6	68.7	71.9	65.5	68.6	84.8	84.3	**87.1**	**85.6**	87.4	91.9	82.8	87.1
20 (Ours)	**77.0**	79.1	**74.9**	**76.9**	**71.0**	**73.8**	**68.3**	**70.9**	84.8	**84.5**	86.1	85.3	**88.7**	**93.0**	**84.5**	**88.5**
30	75.6	**81.2**	70.0	75.2	68.6	72.2	64.9	68.4	**84.9**	**84.5**	86.4	85.4	86.8	91.2	82.5	86.6



(a) epoch 0 (b) epoch 40 (c) epoch 80 (d) epoch 120 (e) epoch 150



(f) epoch 160 (g) epoch 170 (h) epoch 180 (i) epoch 190 (j) epoch 200

Figure 11: t-SNE visualization of our training process on FGVC Aircraft over all classes. The first five subfigures show the GCD-driven representation learning phase, while the remaining five show the Graph Propagation refinement phase.

G Limitations and Future Work

PD-GCD depends on the quality of its graph and on which categories receive queried labels. The CIFAR-100 noise results show that novel-category predictions can deteriorate even when Old accuracy changes little. A fixed confidence threshold may also favor easy categories when selecting pseudo-seeds. An important next step is to connect query allocation with measured graph reliability, so annotation and pseudo-label expansion can focus on poorly supported categories. Adaptive consistency weights offer another direction, motivated by the different loss rankings on Cars, Aircraft, CUB, and CIFAR-100.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction identify the active GCD setting, the acquisition of oracle labels, and the connection to AGCD. The contribution is the use of acquired labels for graph supervision and model refinement. Dataset-level claims follow the current main table.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: The appendix discusses graph quality, sensitivity of novel-category predictions to feature noise, fixed confidence thresholds, and adaptive query allocation and consistency weights.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

   Justification: The appendix provides a harmonic-extension interpretation on an idealized undirected graph. It states symmetry and seed-connectivity conditions and derives the seed-perturbation relation. No new formal performance guarantee is claimed.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper, regardless of whether the code and data are provided or not?

   Answer: [No].

   Justification: The paper describes the method and training procedure; some reproducibility details remain to be completed.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No].

   Justification: The datasets are public and cited. This manuscript source package contains figures and LaTeX files, but it does not include training code or a complete reproduction package.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details, e.g., data splits, hyperparameters, how they were chosen, type of optimizer, necessary to understand the results?

   Answer: [No].

   Justification: The paper specifies the model, class splits, query mechanism, graph settings, losses, and training phases.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No].

   Justification: The tables do not report multi-seed means, standard deviations, confidence intervals, or statistical significance tests.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources, type of compute workers, memory, time of execution, needed to reproduce the experiments?

   Answer: [No].

   Justification: The efficiency tables report measurements on one RTX 4090 for selected benchmark sizes. A complete per-experiment resource and runtime record is not included.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The research uses public image classification benchmarks and does not involve private data, human-subject experiments, surveillance applications, or high-risk deployment. We have reviewed the NeurIPS Code of Ethics and believe the work conforms to it.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: The method may reduce annotation cost and improve open-world recognition by discovering novel categories under limited supervision. Potential risks include misclassification, biased category discovery, and overconfident predictions when deployed in sensitive domains. Such systems should be carefully validated before real-world use.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse, e.g., pre-trained language models, image generators, or scraped datasets?

    Answer: [N/A].

    Justification: The paper does not release high-risk models such as large language models, image generators, or scraped datasets. The experiments are conducted on standard public image classification benchmarks.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets, e.g., code, data, models, used in the paper properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No].

    Justification: The paper cites datasets, pretrained models, and related methods, including AGCD. It does not yet enumerate the license and terms of use for every asset.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No].

    Justification: PD-GCD introduces a method implementation. This source package does not contain its code and reproduction documentation.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation, if any?

    Answer: [N/A].

Justification: The paper does not involve crowdsourcing experiments or research with human subjects. All experiments are conducted on existing public datasets.

15. **Institutional review board approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board approvals, or an equivalent approval/review based on the requirements of your country or institution, were obtained?

Answer: [N/A].

Justification: The paper does not involve human-subject studies, crowdsourcing, or collection of new human data, so IRB approval is not applicable.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A].

Justification: LLM-based tools assisted with manuscript editing, code development and debugging, and data analysis and visualization scripts. LLMs are not a component of the PD-GCD model.