Incremental Generalized Category Discovery – Supplementary Material

Bingchen Zhao Oisin Mac Aodha
University of Edinburgh
https://bzhao.me/iNatIGCD

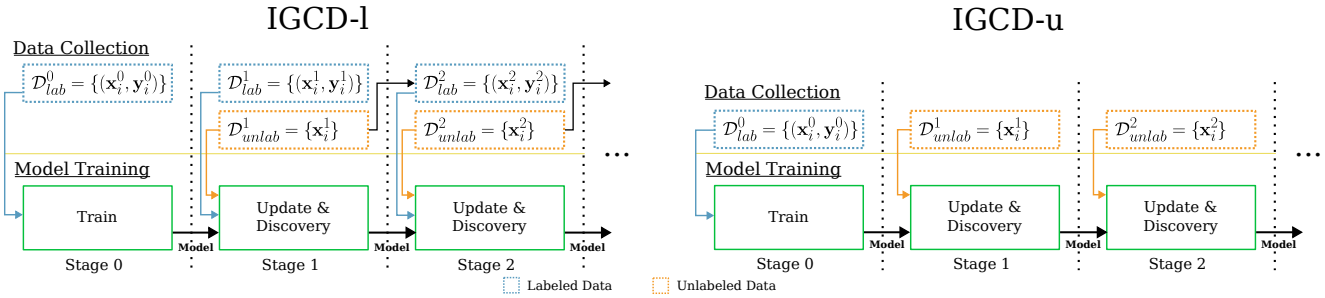

Figure A1: The two incremental learning settings used in our work. In the IGCD-l setting, at each stage we have one unlabeled set and one labeled set. The unlabeled set is annotated at the end of each stage and then used as the labeled set for the next stage. In the IGCD-u setting, only the unlabeled set is provided at each incremental stage, i.e. after the initial label data is given in Stage-0, no additional labels are provided.

A. iNatIGCD Dataset

In Tab. A1, we compare our dataset split to common benchmarks used by previous papers. Our new iNatIGCD benchmark has more categories and images per stage and thus can be used to better evaluate the performance of generalized category discovery methods.

B. Additional Results

B.1. Additional Ablations

Impact of the Size of $\mathcal{R}$. In addition to the ablation on the size of the support set $\mathcal{S}$ in the main paper, here we present results where we vary the size of the replay buffer $\mathcal{R}$. Similar to the ablation of the size of $\mathcal{S}$, we denote the number of images in $\mathcal{R}$ as $N^{\mathcal{R}}$ and the number of categories in $\mathcal{R}$ as $K^{\mathcal{R}}$. The results are presented in Tab. A2. The default value used by our method is illustrated by the gray row, i.e. the second row. We observe a similar trend as Table 6 in the main paper where increasing the number of examples increases the performance. However, the larger number impacts training efficiency especially in the context of the large number of categories in our iNatIGCD dataset.

CLIP pretrained features. We also performed ablation experiments on iNatIGCD-l using CLIP pre-trained feature extractor. In Tab. A3, we can see that due to the challenging fine-grained nature of iNatIGCD, the model failed to perform discovery well. But overall the pre-trained CLIP feature extractor improves the performance, and our method still achieves the best performance.

Density Selection Parameters. There are three hyperparameters to be set in our density selection step described in Sect. 4.3 in the main paper. They are, the number of nearest neighbors K for calculating the density, the number of neighbors K^d, and the threshold T used when removing redundant density peaks. We present the ablation study on these parameters in Tabs. A4 to A6.

In Tab. A4, we vary the value of K. We can see that generally the performance of different K values results in an inverse U shape. As K increases, the performance reaches the best value, and after the performance peak, the performance degrades as K increases further. When K is small, the compared neighborhood is small, thus the estimation of density peaks is not accurate. This results in many noisy peaks that can lead worse performance. When K increases, the density peak estimation is more accurate. However, considering the case where K equals to the number of images in the dataset, we would underestimate the density peaks as many clusters may be considered as one cluster. This would result in a performance decrease when K is higher than bet-

Dataset	Used In	# Classes / Stage	# Stages	# Avg. Images / Stage	# Avg. Classes / Stage
ImageNet-1k	NCDwF [6]	882 / 30	2	39k	30
TinyImageNet	class-iNCD [10]	180 / 20	2	20k	20
CIFAR100-MI	GM [13]	70 / 10 / 10 / 10	4	5k	10
iNatIGCD	Ours	3,888 / 972 / 3,040 / 4,324	4	25k	2.7k

Table A1: The statistics of our dataset compared to previous benchmarks that focus on generalized category discovery under the incremental setting. At Stage-1, 2, and 3, iNatIGCD contains 390, 2,625, and 1,492 novel classes respectively.

$N^{\mathcal{R}}$ / $K^{\mathcal{R}}$	CIFAR-100		iNatIGCD-l	
	$M_f \downarrow$	$M_d \uparrow$	$M_f \downarrow$	$M_d \uparrow$
1	25.1	18.6	38.2	15.8
3	4.6	29.2	28.3	27.1
5	4.0	31.2	28.1	26.6
7	4.4	34.2	27.1	27.9
10	**3.4**	**36.1**	**24.0**	**29.4**

Table A2: Impact of the size of the replay buffer $\mathcal{R}$.

$M_f \downarrow / M_d \uparrow$	CLIP-RN-50	RN-18	ViT-B
GM [13]	28.4 / 25.6	30.0 / 20.1	**29.1** / 24.8
Ours	**25.4 / 34.2**	**27.6 / 28.4**	29.4 / **30.2**

Table A3: Performance of using CLIP pretrained features on iNAT-IGCD-l dataset.

ter choices.

K	CIFAR-100		iNatIGCD-l	
	$M_f \downarrow$	$M_d \uparrow$	$M_f \downarrow$	$M_d \uparrow$
5	4.8	27.1	28.9	27.1
10	4.3	28.9	27.9	26.9
15	4.5	**29.7**	26.1	**27.8**
20	4.1	28.1	**25.9**	27.6
40	**3.8**	26.4	27.2	26.0

Table A4: Ablation of K used in density-based selection.

In Tab. A5, we ablate the choice of K^d. This hyper-parameter is used to remove redundant density peaks. When the value of K^d is low, it would remove fewer density peaks as two peaks' K^d neighbor set are less likely to overlap. When the value is high, it would remove more density peaks. From the results, we can see that setting K^d to a higher number than K (which equals 10 in Tab. A5) can result in better performance.

Another hyper-parameter in density selection is the threshold T. Similar to the selection of K^d, a higher threshold will remove fewer density peaks, while a lower threshold will remove more. The results are presented in Tab. A6. Our choice from the main paper is shaded in gray, which obtains a good balance between forgetting and discovery.

K^d	CIFAR-100		iNatIGCD-l	
	$M_f \downarrow$	$M_d \uparrow$	$M_f \downarrow$	$M_d \uparrow$
10	4.1	20.3	28.0	18.9
15	4.8	23.2	27.5	23.4
20	4.3	28.9	27.9	26.9
30	**3.8**	**29.5**	27.1	**27.9**
40	4.5	28.1	**26.9**	27.2

Table A5: Ablation of K^d used in density-based selection.

T	CIFAR-100		iNatIGCD-l	
	$M_f \downarrow$	$M_d \uparrow$	$M_f \downarrow$	$M_d \uparrow$
0.2	5.1	28.7	29.2	**27.0**
0.4	4.8	**29.3**	28.9	26.8
0.6	4.3	28.9	27.9	26.9
0.8	**4.0**	27.2	**27.1**	26.0

Table A6: Ablation of T used in density-based selection.

B.2. Vision Transformer Results

In the main paper, our results are presented using a ResNet-18 backbone. In Tabs. A7 and A8 we present results using a vision transformer [3] Base 16 model. We observe that the overall trend remains the same. The vision transformer results in an increase in performance compared to the ResNet-18 [5] that we use in the main paper, but at the cost of being much slower to train.

B.3. CIFAR-100 IGCD-l and IGCD-u Results

In Tabs. A9 and A10, we re-purpose the CIFAR-100 dataset for the IGCD-l and IGCD-u settings, using the same category split as [13], and present the results comparing our method with previous SoTA baselines. We can see that our method, when compare with others, still achieves a competitive performance. In terms of the overall performance metrics $\mathcal{M}_f$ and $\mathcal{M}_d$, our method achieves the best results for both CIFAR-100-IGCD-l and CIFAR-100-IGCD-u.

B.4. Results on Static GCD Datasets

In Tab. A11 we present results on the static GCD benchmarks, i.e. without any incremental stages. For the adaptation of our method to the static GCD scenario, we remove the incremental update of $\mathcal{S}$ and $\mathcal{R}$. We can see that our

Methods	Stage-0	Stage-1			Stage-2				Stage-3					Overall	
	All	All	Old	New	All	Old	New	S-0	All	Old	New	S-1	S-0	M_f	M_d
SimGCD + iCaRL [9]	45.6	25.2	33.6	14.1	24.7	**38.0**	16.9	9.7	24.0	37.0	17.8	15.3	9.7	35.9	26.9
GM [13]	**46.0**	19.5	28.1	9.4	20.9	29.6	11.2	16.2	18.3	27.1	16.8	17.0	14.3	**29.1**	24.8
Ours	45.6	**27.0**	**36.3**	**16.7**	**26.8**	37.2	**23.4**	**18.5**	**25.0**	**38.1**	**19.7**	**18.2**	**17.1**	29.4	**30.2**

Table A7: Vision transformer-backbone results on iNatIGCD in the IGCD-l setting (i.e. where labels are available at the end of each stage). Higher numbers are better, with the exception of M_f, where lower is better.

Methods	Stage-0	Stage-1		Stage-2			Stage-3				Overall	
	All	New	S-1	New	S-1	S-0	New	S-2	S-1	S-0	M_f	M_d
SimGCD + iCaRL [9]	42.0	9.8	17.9	14.6	8.9	12.6	15.7	11.2	7.4	10.4	31.3	12.6
GM [13]	**42.1**	9.0	18.0	14.0	9.4	**15.4**	10.8	11.3	8.4	13.1	29.2	10.6
Ours	41.9	**12.8**	**22.5**	**15.7**	**10.9**	14.6	**16.3**	**11.5**	**9.0**	**13.6**	**28.4**	**14.9**

Table A8: Vision transformer-backbone results on iNatIGCD in the IGCD-u setting (i.e. where labels are not available at the end of each stage). Higher numbers are better, with the exception of M_f, where lower is better.

proposed method still performs on par with previous SoTA methods on static GCD benchmarks despite primarily being designed for the incremental setting.

C. Implementation Details

C.1. Training Losses

We describe the representation learning losses $\mathcal{L}_{\text{rep}}$ below which follows a contrastive learning framework. Formally, given two views (i.e. augmentations) $\hat{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_i$ of an input image $\boldsymbol{x}_i$ in a mini-batch $\mathcal{B}$, the self-supervised component of the contrastive loss can be written as

$$\mathcal{L}_{\text{SelfCon}} = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} -\log \frac{\exp\left(\hat{\boldsymbol{z}}_i^\top \tilde{\boldsymbol{z}}_i'/\tau_u\right)}{\sum_i^{i\neq n}\exp\left(\hat{\boldsymbol{z}}_i^\top \tilde{\boldsymbol{z}}_n'/\tau_u\right)}, \quad (1)$$

where the embedding $\boldsymbol{z} = m(f(\boldsymbol{x}))$ is a projected feature from a MLP projector m as in [2, 12], and τ_u is a temperature parameter. The supervised contrastive loss [7] is similar with the difference being that the positive samples are matched with their ground truth labels,

$$\mathcal{L}_{\text{SupCon}} = \frac{1}{|\mathcal{B}^l|}\sum_{i\in\mathcal{B}^l}\frac{1}{|\mathcal{M}_i|}\sum_{q\in\mathcal{M}_i} -\log \frac{\exp\left(\hat{\boldsymbol{z}}_i^\top \tilde{\boldsymbol{z}}_q/\tau_c\right)}{\sum_n^{n\neq i}\exp\left(\hat{\boldsymbol{z}}_i^\top \tilde{\boldsymbol{z}}_n/\tau_c\right)}, \quad (2)$$

where $\mathcal{M}_i$ indexes all other images in the same batch that have the same label as $\boldsymbol{x}_i$, and τ_c is the temperature parameter for the supervised contrastive loss.

For the fully supervised upper bound discussed in Tabs. 1 and 2 in the main paper, we leverage both contrastive learning losses $\mathcal{L}_{\text{SelfCon}}$ and $\mathcal{L}_{\text{SupCon}}$ for representation learning. For the classifier, we simply adopt the cross-entropy loss $\mathcal{L}_{\text{ce}}$ on all examples. We concatenate all datasets $\mathcal{D}^t_{lab}, \mathcal{D}^t_{unlab}$ from every time step t, and provide the ground truth category labels for the unlabelled datasets $\mathcal{D}^t_{unlab}$. The resulting concatenated dataset is used to train the fully supervised baseline. So in this setting, the model will have labels for all the categories and no forgetting occurs. This can be considered as a very strong upper bound on the performance for our IGCD setting.

```
# feats: feature vectors of the unlabeled set (NxC).
# supps: existing support samples, normed (SxC)
# K: the number of neighbours to use for density
# Kd: the number of neighbours to use for IoU
# thres: threshold for IoU and NMS
# returns selected_peaks

# get k nearest neighbors
normed_feats = norm(feats)
sims = normed_feats @ normed_feats.t()
knn_indice = topk(sims, K)

# get density peaks
density = sims[knn_indice].sum() / K
nn_density = density[knn_indice]
peaks_ind = [i where density[i] > nn_density[i]]

# use iou on nns to dedup
knn_ind = topk(sims, k=Kd)
peak_nns = knn_ind[peaks_ind]
selected_peaks = [peaks_ind[0]]
for peak, nn in zip(peaks_ind[1:], peak_nns):
    flag = True
    for other_nn in peak_nns.remove(nn):
        if IoU(nn, other_nn) > thres:
            flag = False
    if flag:
        selected_peaks.append(peak)
```

Listing 1: Pseudo-code of our density selection step.

C.2. Density Selection

We present the pseudo-code for our proposed density selection process in Listing 1. We use ℓ_2-normed features to calculate the density of the examples, and then filter out the density peaks by comparing the density of one node to its k

Methods	Stage-0	Stage-1			Stage-2				Stage-3					Overall	
	All	All	Old	New	All	Old	New	S-0	All	Old	New	S-1	S-0	M_f	M_d
Supervised upper-bound	79.2	82.5	80.3	85.3	81.0	82.4	80.0	79.2	81.3	82.3	80.2	81.7	80.2	-	-
SimGCD + iCaRL [9]	77.8	**78.6**	**79.6**	76.7	77.9	78.6	74.5	63.4	75.6	**78.9**	70.8	60.2	50.1	27.7	67.8
GM [13]	79.0	78.3	78.8	77.3	**78.2**	**78.9**	74.7	**66.3**	**76.7**	77.8	71.2	63.2	49.8	29.2	68.3
Ours	77.8	78.5	79.0	78.2	78.0	78.7	**75.0**	65.3	76.0	78.8	**72.1**	**64.6**	**51.3**	**26.5**	**69.2**

Table A9: Results on CIFAR-100 in the IGCD-l setting (i.e. where labels are available at the end of each stage). Higher numbers are better, with the exception of M_f, where lower is better.

Methods	Stage-0	Stage-1		Stage-2			Stage-3				Overall	
	All	New	S-0	New	S-1	S-0	New	S-2	S-1	S-0	M_f	M_d
Supervised upper-bound	79.2	78.9	81.2	80.2	79.9	81.4	85.3	82.1	80.1	79.0	-	-
SimGCD + iCaRL [9]	77.8	62.4	43.2	66.7	40.8	38.7	**68.2**	41.3	36.2	35.3	42.5	50.1
GM [13]	79.0	58.9	45.3	63.4	46.8	**43.2**	60.0	**42.5**	45.6	40.2	38.8	53.4
Ours	77.8	**64.4**	**46.9**	**67.8**	**47.0**	42.1	68.0	40.0	**46.8**	**41.5**	**36.3**	**55.9**

Table A10: Results on CIFAR-100 in the IGCD-u setting (i.e. where labeled data is not provided during the incremental stages). Higher numbers are better, with the exception of M_f, where lower is better.

Methods	ImageNet-100			SCars		
	All	Old	New	All	Old	New
GCD [11]	74.1	89.8	66.3	39.0	57.6	29.9
ORCA [1]	73.5	**92.6**	63.9	23.5	50.1	10.7
SimGCD [12]	82.4	90.7	78.3	46.8	**64.9**	38.0
Ours	**83.0**	89.5	**79.1**	**47.2**	63.8	**38.7**

Table A11: Results on static GCD benchmarks.

nearest neighbours. Then we compare each pair of density peaks and remove peaks that have nearest neighbours set to overlap with other peaks larger than a threshold.

C.3. Evaluation Metrics

The main results in each of the tables and figures are reported using the clustering accuracy (ACC). At each evaluation stage, given the ground truth $\boldsymbol{y}$ and a predicted label $\hat{\boldsymbol{y}}$, the ACC is calculated as $\text{ACC} = \frac{1}{M}\sum_{i=1}^{M}\mathbb{1}(\boldsymbol{y}_i = p(\hat{\boldsymbol{y}}_i))$. Here, $M = |\mathcal{D}^u|$ and p is the optimal permutation that matches the predicted cluster assignments to the ground truth category labels. Under the IGCD-l setting, we report the ACC for all the categories in Stage-0 as all categories have labeled training examples. For later stages, we report the 'All', 'Old', and 'New' performance which correspond to all categories, the labeled categories, and the unlabeled categories at each stage. Additionally, we use S-t to denote the categories that appear in a previous stage t but are not presented in the current stage to measure the forgetting of the model. Finally, as noted in the main paper, we also use the summary metrics defined in [13] to measure overall discovery and forgetting performance, $\mathcal{M}_d$ and $\mathcal{M}_f$.

C.4. Experimental Settings

Our models are trained using a stochastic gradient descent optimizer with an initial learning rate of 0.1, momentum of 0.9, and weight decay of $1e-4$. The learning rate is decayed following a cosine schedule [8]. We use a batch size of 128 during training for all datasets, where 64 images are sampled from $\mathcal{D}^t_{lab}$ and 64 images are sampled from $\mathcal{D}^t_{unlab}$. The balancing factor λ_{rep} is set to 0.35, ϵ is set to 2.0, K is set to 10, K^d is set to 20, and the threshold T is set to 0.6. We adopt the same set of augmentations used in [4] for contrastive learning. The temperature parameters in contrastive learning τ_u and τ_c are set to 0.07 and 0.1 respectively.

References

[1] Kaidi Cao, Maria Brbić, and Jure Leskovec. Open-world semi-supervised learning. In *ICLR*, 2022. 4

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 3

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 2

[4] Enrico Fini, Enver Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. A unified objective for novel class discovery. In *ICCV*, 2021. 4

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[6] KJ Joseph, Sujoy Paul, Gaurav Aggarwal, Soma Biswas, Piyush Rai, Kai Han, and Vineeth N Balasubramanian.

Novel class discovery without forgetting. In *ECCV*, 2022. 2

[7] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *NeurIPS*, 2020. 3

[8] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. 4

[9] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 3, 4

[10] Subhankar Roy, Mingxuan Liu, Zhun Zhong, Nicu Sebe, and Elisa Ricci. Class-incremental novel class discovery. In *ECCV*, 2022. 2

[11] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Generalized category discovery. In *CVPR*, 2022. 4

[12] Xin Wen, Bingchen Zhao, and Xiaojuan Qi. Parametric classification for generalized category discovery: A baseline study. In *ICCV*, 2023. 3, 4

[13] Xinwei Zhang, Jianwen Jiang, Yutong Feng, Zhi-Fan Wu, Xibin Zhao, Hai Wan, Mingqian Tang, Rong Jin, and Yue Gao. Grow and merge: A unified framework for continuous categories discovery. In *NeurIPS*, 2022. 2, 3, 4