Supplementary Material of Decouple Your Discovery and Memory in Continual Generalized Category Discovery

Jiawei Yu[1,2]†, Zijian Gao[1,2]†, Xingxing Zhang[3], Xuan Liu[4], Huaimin Wang[1,2], Kele Xu[1,2]*
[1]College of Computer Science and Technology, National University of Defense Technology
[2]State Key Laboratory of Complex & Critical Software Environment
[3]School of Computer Science, Tsinghua University
[4]College of Computer Science and Electronic Engineering, Hunan University
{yujiawei23,gaozijian19,hmwang,xukelele}@nudt.edu.cn,
xxzhang1993@gmail.com, xuan_liu@hnu.edu.cn

1. Proof

Proof. In the main text, at stage t, we have the learning problem

$$\underset{\boldsymbol{W}_t^{\mathrm{L}}}{\operatorname{argmin}} \left\|\boldsymbol{Y}_{0:t} - \boldsymbol{X}_{0:t}^{\mathrm{E}} \boldsymbol{W}_t^{\mathrm{L}}\right\|_F^2 + \gamma \left\|\boldsymbol{W}_t^{\mathrm{L}}\right\|_F^2,$$

which leads to optimal estimation

$$\hat{\boldsymbol{W}}_t^{\mathrm{L}} = \left(\boldsymbol{X}_{0:t}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{X}_{0:t}^{\mathrm{E}} + \gamma \boldsymbol{I}\right)^{-1} \boldsymbol{X}_{0:t}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t} = \boldsymbol{R}_t \boldsymbol{X}_{0:t}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t}. \tag{1}$$

From the definition of the auto-correlation matrix, we can get the equation

$$\boldsymbol{R}_t = \left(\boldsymbol{R}_{t-1}^{-1} + \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{X}_t^{\mathrm{E}}\right)^{-1}. \tag{2}$$

According to the Woodbury matrix identity, we can complete the proof for the recursive formulation of $\boldsymbol{R}_t$ (i.e. equation 15 in the main text) as

$$\boldsymbol{R}_t = \boldsymbol{R}_{t-1} - \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1}. \tag{3}$$

We can break the estimation 1 into

$$\begin{aligned} \hat{\boldsymbol{W}}_t^{\mathrm{L}} &= \boldsymbol{R}_t \begin{bmatrix} \boldsymbol{X}_{0:t-1}^{\mathrm{E}}{}^{\mathrm{T}} & \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{Y}_{0:t-1} & 0 \\ 0 & \boldsymbol{Y}_t \end{bmatrix} \\ &= \left[\underbrace{\boldsymbol{R}_t \boldsymbol{X}_{0:t-1}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t-1}}_{\text{the first term}} \quad \underbrace{\boldsymbol{R}_t \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_t}_{\text{the second term}} \right]. \end{aligned} \tag{4}$$

The first term obviously involves the historical data, while the second term does not. We aim to deal with it with the weight $\hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}}$ of stage $t-1$. Combing equation 3, we can rewrite the first term as

$$\begin{aligned} &\boldsymbol{R}_t \boldsymbol{X}_{0:t-1}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t-1} \\ &= \boldsymbol{R}_{t-1} \boldsymbol{X}_{0:t-1}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t-1} - \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \\ &\left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_{0:t-1}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t-1} \\ &= \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} - \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \boldsymbol{X}_t^{\mathrm{E}} \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} \end{aligned} \tag{5}$$

Taking $\boldsymbol{B_t} = \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1}$, we have

$$\begin{aligned} \boldsymbol{I} = \boldsymbol{B_t} \boldsymbol{B_t}^{-1} &= \boldsymbol{B_t}\left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right) \\ &= \boldsymbol{B_t} + \boldsymbol{B_t}\left(\boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right) \\ &= \boldsymbol{B_t} + \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \left(\boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right) \end{aligned}$$

$\boldsymbol{B_t} = \boldsymbol{I} - \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \left(\boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)$, thus we obtain. Based on this, we take advantage of the recursive formulation $\boldsymbol{R}_t$ in equation 3 and rewrite equation 5 into

$$\begin{aligned} &\boldsymbol{R}_t \boldsymbol{X}_{0:t-1}^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{Y}_{0:t-1} \\ &= \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} - \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \\ &\left(\boldsymbol{I} - \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right) \boldsymbol{X}_t^{\mathrm{E}} \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} \\ &= \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} - \\ &\left(\boldsymbol{R}_{t-1} - \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \left(\boldsymbol{I} + \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1} \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}}\right)^{-1} \boldsymbol{X}_t^{\mathrm{E}} \boldsymbol{R}_{t-1}\right) \\ &\boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{X}_t^{\mathrm{E}} \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} \\ &= \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}} - \boldsymbol{R}_t \boldsymbol{X}_t^{\mathrm{E}}{}^{\mathrm{T}} \boldsymbol{X}_t^{\mathrm{E}} \hat{\boldsymbol{W}}_{t-1}^{\mathrm{L}}. \end{aligned}$$

*Corresponding author. † Equal Contribution.

Algorithm 1 Training Algorithm of DYDM

Input: Pre-trained encoder f; number of epochs E; regularization hyperparameter γ
Data: Sequence data of incremental stages $\mathcal{D}_0, \cdots, \mathcal{D}_n$
Output: Feature encoder f_0, expansion layer f_E and linear layer f_L for inference.
Initialize classification head g; expansion layer f_E; linear layer f_L
for $t = 0$ to n **do**
  if $t = 0$ **then**
    for $e = 1$ to E **do**
      Sample batch $\{x_0, y_0\}$ from $\mathcal{D}_0$
      Optimize f_0 and g_0 by minimizing $\mathcal{L}_{\text{base}}$
    Sample all data $\{\boldsymbol{X}_0, \boldsymbol{Y}_0\}$ from $\mathcal{D}_0$
    Get expanded feature matrix $\boldsymbol{X}_0^{\text{E}}$ using Eq.6
    Calculate weight matrix $\boldsymbol{W}_0^{\text{L}}$ of f_L using Eq.8
    Calculate the auto-correlation matrix $\boldsymbol{R}_0 = \left(\boldsymbol{X}_0^{\text{E}^{\text{T}}} \boldsymbol{X}_0^{\text{E}} + \gamma \boldsymbol{I}\right)^{-1}$.
  else
    for $e = 1$ to E **do**
      Sample batch x_t from $\mathcal{D}_t$
      Optimize f_t and g_t by minimizing $\mathcal{L}_{\text{new}}$ and $\mathcal{L}_{\text{KD}}$
    Sample all data $\boldsymbol{X}_t$ from $\mathcal{D}_t$
    Calculate indicator matrix $\boldsymbol{M}$ using Eq.9
    Construct novel class set $\hat{\boldsymbol{X}}_t$ and one-hot pseudo label $\hat{\boldsymbol{Y}}_t$ using Eq.10
    Get expanded feature matrix $\hat{\boldsymbol{X}}_t^{\text{E}}$ using Eq.6
    Calculate the auto-correlation matrix $\boldsymbol{R}_t$ using Eq.15
    Calculate weight matrix $\boldsymbol{W}_t^{\text{L}}$ of f_L using Eq.14

Hence, we can complete the proof and rewrite the estimation 4:

$$\hat{\boldsymbol{W}}_t^{\text{L}} = \left[\underbrace{\hat{\boldsymbol{W}}_{t-1}^{\text{L}} - \boldsymbol{R}_t \boldsymbol{X}_t^{\text{E}^{\text{T}}} \boldsymbol{X}_t^{\text{E}} \hat{\boldsymbol{W}}_{t-1}^{\text{L}}}_{\text{all historical data}} \quad \underbrace{\boldsymbol{R}_t \boldsymbol{X}_t^{\text{E}^{\text{T}}} \boldsymbol{Y}_t}_{\text{current data}} \right].$$

At stage t, we can recursively obtain weight matrix $\hat{\boldsymbol{W}}_t^{\text{L}}$ using only current data, auto-correlation matrix $\boldsymbol{R}_{t-1}$ and the previous stage weight $\boldsymbol{W}_{t-1}^{\text{L}}$, the latter two representing all historical data. □

2. Algorithm of the Proposed Method

Please refer to Algorithm 1.

3. Implementation Details

Dataset Split Details. We present the detailed splits of the four standard datasets: CIFAR100 (C100), Tiny-ImageNet (Tiny), ImageNet-100 (IN100) and CUB200 in Table 1. Notably, for the CGCD task, both the number and order of classes learned at each stage are crucial.

Table 1. Dataset splits of C-GCD setting. The symbols # and / denote "number" and "per" respectively.

Dataset	Stage-0		Each Stage-t (t = 1, ⋯, 25)		
	#base class	#img/base class	#new class	#img/new class	#img/old class
CIFAR100	50	400	50/t	400	25
ImageNet-100	50	∼1,000 (80%)	50/t	1,000	60
Tiny-ImageNet	100	400	100/t	400	25
CUB200	100	∼25 (80%)	100/t	25	5

Training details. During the initial stage, the model is trained for 100 epochs using SGD with momentum (0.9), a learning rate of 0.1, and a weight decay of 5e-5, with early stopping applied to prevent overfitting. In the continual discovery stages, the learning rate is set to 0.01 and training lasts for 30 epochs. A cosine annealing schedule is applied throughout. The batch size is fixed at 128. All experiments are conducted on a NVIDIA GeForce RTX 4090 GPU.

Model details. We adopt a ViT-B/16 backbone with DINO pretraining and use the output [CLS] token as the feature representation. For all previous methods and the discovery branch in our approach, we finetune the last transformer block. Following the DINO framework, we employ a parametric classifier composed of a multilayer perceptron (MLP) and a bias-free, weight-normalized linear layer. The MLP receives a 768-dimensional feature vector (the ViT backbone output) and projects it into a 256-dimensional

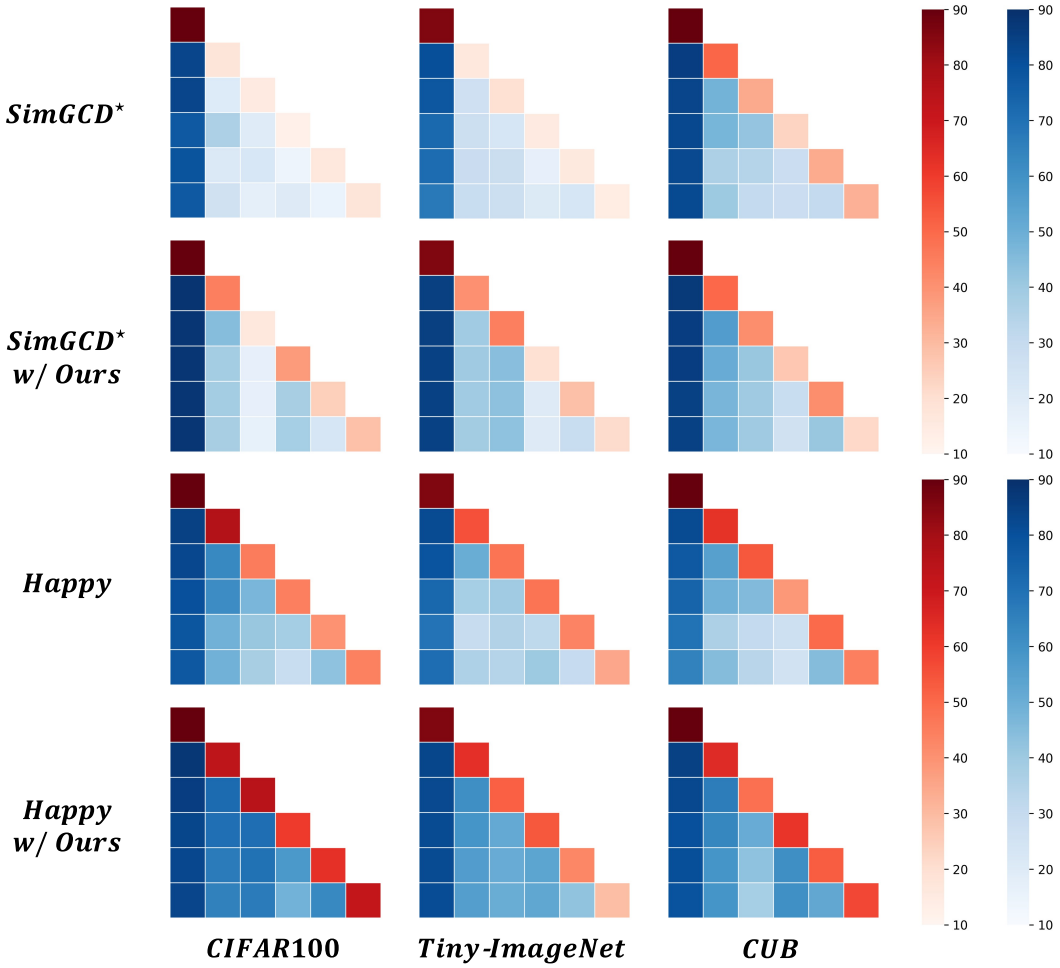

Figure 1. Performance heatmaps of SimGCD*, Happy, and the play-and-plug performance.

bottleneck space through three linear transformations interleaved with two GELU activations. Specifically, the first layer maps from 768 to 2048 dimensions, followed by a GELU activation. The second layer maintains the 2048-dimensional space with another GELU. Finally, a linear layer reduces the dimensionality to 256. All MLP weights are initialized using a truncated normal distribution with a standard deviation of 0.02, and biases are initialized to zero. Additionally, we use a weight-normalized linear classifier with shape $768 \times c$, where c is the number of classes. Input features are ℓ_2-normalized before being passed to this layer, encouraging learning on the hypersphere. Following the DINO strategy, the weight scaling factor g is fixed such that the weight norm equals 1.0. This classifier serves as the output layer in both the initial stage and the discovery branch. The analytic classifier in the memory branch adopts a lightweight two-layer perceptron as its classification head. It consists of two fully connected layers with a ReLU activation in between. Specifically, the input feature vector is first projected into a hidden dimension, followed by a ReLU activation, and then linearly transformed to the final output dimension corresponding to the number of classes. Both linear layers are implemented without bias terms.

4. Extended Results

This section provides some extended experimental results for the main text. These results offer deeper insights into the effectiveness and generalizability of our proposed method under various settings.

4.1. Extended Plugin Analysis

To intuitively assess the performance of C-GCD on both previously learned and newly introduced classes, we present a heatmap visualization of a lower-triangular accuracy matrix (Figure 3). In this matrix, each row corresponds to a training stage S_i, and each entry A_{ij} denotes the prediction accuracy on task S_j after training up to stage S_i, where $j \leq i$.

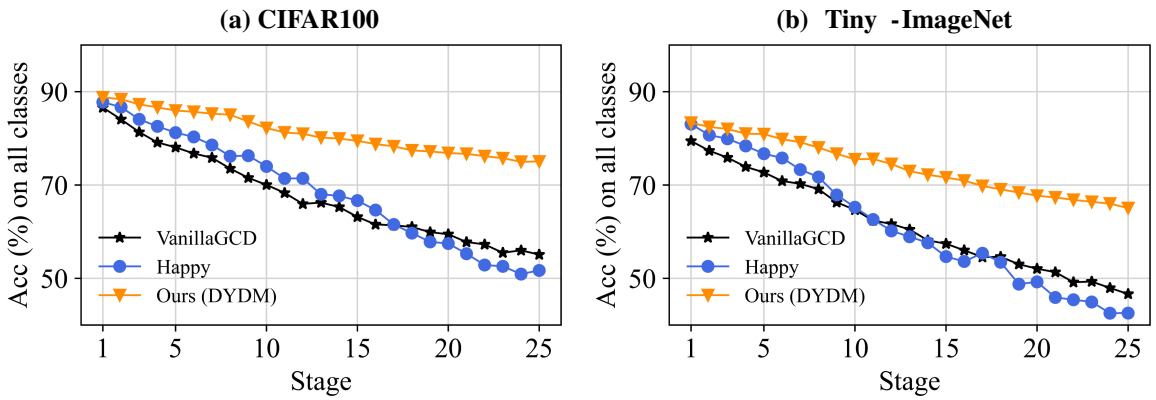

Figure 2. Performance of 25 continual stages on CIFAR100 and Tiny-ImageNet.

Color intensity represents the accuracy value: blue regions indicate knowledge retention on earlier tasks, while red regions reflect the model's ability to discover and learn new categories. This visualization provides insight into how well the model maintains a balance between stability (preserving prior knowledge) and plasticity (accommodating new information) throughout the continual discovery process.

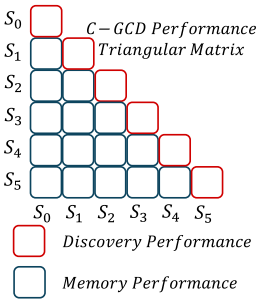

Figure 3. C-GCD Performance Matrix.

As shown in Figure 1, we use the heatmap to comprehensively demonstrate the plug-in capability of our proposed memory branch. We conduct experiments on three representative datasets: CIFAR-100, Tiny-ImageNet, and CUB. It can be observed that for weak baseline methods like SimGCD*, which exhibits poor anti-forgetting and category discovery capabilities (manifested as lighter blue and red regions in the heatmap), and for strong baseline methods like Happy, which already performs exceptionally well in memory retention and novel knowledge acquisition (manifested as darker blue and red regions in the heatmap), the addition of the memory branch brings further performance improvements. This observation underscores the modular compatibility of our proposed design, which can be seamlessly integrated into a variety of existing architectures. Notably, regardless of the significant performance gaps among the base methods, the memory branch consistently brings measurable improvements. Moreover, the introduction of the memory branch substantially enhances the model's capacity to retain knowledge from previously learned tasks, while simultaneously improving its ability to recognize new categories. These findings further validate that our approach effectively balances the trade-off between stability and plasticity, thereby demonstrating the efficacy of the decoupled architecture within the continual generalized category discovery framework.

Table 2. The influence of random expansion layers.

Dataset	Randomization	All	Old	New
CIFAR100	✓	73.79	74.91	63.70
	×	71.66	73.03	59.30
CUB	✓	65.65	68.52	40.07
	×	63.15	66.91	29.62
Tiny-ImageNet	✓	63.35	65.59	43.20
	×	61.21	64.12	35.00

4.2. Performance on Long-stage Task Setting

We evaluate the performance of three category discovery methods across 25 incremental discovery stages on CIFAR-100 and Tiny-ImageNet. As shown in Table 4 and Table 5, we present the stage-wise evolution of overall, old class, and new class accuracies from top to bottom. In addition, we report the average accuracy of each metric across all 25 stages, where our method consistently achieves the best performance. The visualization of the overall performance in Figure 2 shows our significant improvement.

In terms of new class performance, different methods exhibit substantial fluctuations across the 25-stage setting. This variability can be attributed to the fact that, as the number of stages increases, each stage introduces fewer novel classes, thereby increasing the sensitivity of the discovery process to the method's capability in handling task difficulty. Some approaches, such as Happy and the Vanilla GCD, demonstrate pronounced sensitivity to these changes and even experience severe degradation in certain phases, at times nearly failing to discover any new class at all (as highlighted by the bordered entries in Table 4 and Table 5). In contrast, our method maintains a robust and sta-

Table 3. Comparison of computational efficiency and storage costs across different datasets and tasks.

Dataset – Task	Method	Computational Cost			Storage Cost			
		Training Time/Batch (s)	Inference Time (s)	FPS	Expansion Params	Model Params	Trainable Params	FLOPs
CIFAR100 – Task 1	Ours	33.36	7.08	846.96	2.30M	94.47M	15.91M	4.318T
	Happy	58.16	7.24	828.97	–	91.99M	13.43M	4.319T
CIFAR100 – Task 5	Ours	49.47	11.79	848.01	2.30M	94.62M	16.06M	4.318T
	Happy	68.81	11.81	846.70	–	92.02M	13.46M	4.319T
CUB – Task 1	Ours	7.73	4.09	845.95	2.30M	94.70M	16.14M	4.318T
	Happy	14.70	4.09	843.98	–	92.04M	13.48M	4.319T
CUB – Task 5	Ours	9.71	6.86	844.64	2.30M	95.00M	16.44M	4.318T
	Happy	18.88	6.89	840.93	–	92.10M	13.54M	4.319T

ble new-class discovery performance throughout. These results demonstrate that, even under the challenging setting of long-stage discovery, our method effectively mitigates forgetting while maintaining stable performance in discovering new categories. This highlights its strong adaptability and robustness, indicating promising potential for more complex real-world applications.

4.3. The Benefit of Randomization Techniques.

The expansion layer is a random linear layer that projects features into a higher-dimensional space. This projection increases the capacity of the analytic model by enabling richer and more expressive representations. Prior works, such as A Pseudoinverse Learning Algorithm for Feedforward Neural Networks with Stacked Generalization Applications to Software Reliability Growth Data, Correlation Projection for Analytic Learning of a Classification Network, and Ranpac: Random Projections and Pre-trained Models for Continual Learning, have shown that randomization techniques can help capture useful patterns, improve linear separability, and enhance performance by expanding the feature space without introducing additional training costs. To further validate its effectiveness, in Table 2, we provide performance comparisons of our method with and without randomization. The results across three datasets show consistent performance gains, particularly in novel class accuracy, confirming the benefit of this design.

4.4. Computational Costs Comparisons.

We report efficiency metrics under the 5-stage setting in Table 3, where each phase contains more classes and samples than in the 25-stage setup, while the total number of accumulated classes remains the same. We compare DYDM with the state-of-the-art method Happy in terms of training time per epoch (seconds), inference latency on the full test set (milliseconds), parameter count (millions), FLOPs (teraFLOPs), and throughput measured by frames per second (FPS). DYDM achieves an average inference latency of 1.18 milliseconds and maintains a low number of trainable parameters that grow slowly as new classes are introduced. It also exhibits lower inference-time FLOPs compared to Happy. In addition, while Happy relies on replay and feature distillation to mitigate forgetting, DYDM achieves higher performance with lower training cost through its dual-branch design. Notably, our added components—the frozen random expansion layer and the analytic classifier—are extremely lightweight. The analytic classifier is updated via matrix operations over a single epoch without GPU-based backpropagation. The increase in trainable parameters from Happy to DYDM is minimal, rising from 13.46M to 16.06M (on CIFAR100 Task 5), accounting for only 2.83% of the total model size. This overhead is marginal and falls within an acceptable range. These results demonstrate that DYDM is efficient, scalable, and well-suited for long-term deployment.

5. Future Discussion

Although continual generalized category discovery has progressed in addressing the challenge of classifying both known and novel categories from continuously incoming unlabeled data, current research has primarily focused on the domain of natural images. However, we argue that extending C-GCD to specialized domains such as medical imaging and materials science presents a highly promising and underexplored direction. These fields inherently exhibit characteristics such as a scarcity of labeled data, open-set conditions, and evolving category definitions—for instance, the continual emergence of new disease types or novel material microstructures, as well as the redefinition of existing categories over time. These properties closely align with the core paradigm of C-GCD. For example, in medical imaging, C-GCD has the potential to facilitate the automatic dis-

Table 4. 25-Stage Performance on CIFAR100.

Methods	Stage	1	2	3	4	5	6	7	8	9	10
VanillaGCD	All	86.60	84.07	81.34	79.12	78.07	76.76	75.83	73.53	71.54	70.03
	Old	89.24	86.35	81.76	80.68	78.76	78.03	77.50	75.52	73.36	71.38
	New	20.50	25.00	70.00	35.50	58.00	38.50	24.00	10.00	11.50	24.00
Happy	All	87.75	86.70	84.07	82.57	81.22	80.31	78.58	76.18	76.28	73.99
	Old	87.56	86.62	83.72	81.98	80.74	79.93	78.16	75.58	77.27	75.63
	New	92.50	89.00	93.50	99.00	95.00	91.50	91.50	95.50	43.50	18.00
DYDM (Ours)	All	88.79	88.37	87.27	86.64	85.95	85.68	85.28	85.05	83.60	82.21
	Old	88.88	88.27	87.44	86.50	86.07	85.68	85.32	85.19	84.76	82.40
	New	86.50	91.00	82.50	90.50	82.50	85.50	84.00	80.50	45.50	76.00

Methods	Stage	11	12	13	14	15	16	17	18	19	20
VanillaGCD	All	68.31	65.95	66.22	65.28	63.16	61.55	61.36	61.00	59.89	59.44
	Old	69.57	66.47	67.00	66.11	63.79	62.74	61.76	61.81	60.29	60.31
	New	24.00	47.00	37.50	34.00	38.50	14.00	45.00	27.00	42.50	21.50
Happy	All	71.40	71.42	68.00	67.68	66.66	64.63	61.54	59.69	57.81	57.46
	Old	71.91	72.03	69.82	68.25	67.15	65.19	61.84	61.10	58.08	57.66
	New	53.50	49.50	0.50	46.00	47.50	42.50	49.00	0.50	46.00	48.50
DYDM (Ours)	All	81.25	80.97	80.16	79.95	79.46	78.76	78.29	77.41	77.19	76.83
	Old	82.06	80.86	80.73	79.78	79.41	79.11	78.24	77.85	77.24	76.80
	New	53.00	85.00	59.00	86.50	81.50	64.50	80.00	59.00	75.00	78.50

Methods	Stage	21	22	23	24	25	Avg ACC
VanillaGCD	All	57.78	57.26	55.48	55.99	55.08	67.63
	Old	58.53	57.92	56.22	56.48	55.06	68.67
	New	24.00	26.50	20.50	32.50	56.00	32.30
Happy	All	55.28	52.85	52.57	50.92	51.69	68.69
	Old	54.37	52.99	51.63	50.94	50.71	68.83
	New	96.50	46.50	97.00	50.00	99.50	63.28
DYDM (Ours)	All	76.68	76.17	75.74	74.97	75.03	**81.11**
	Old	76.56	76.33	75.73	75.53	74.61	**81.25**
	New	82.50	69.00	76.00	48.00	95.50	**75.90**

covery of novel pathological patterns. Similarly, in materials science, advancements in experimental techniques and data acquisition methods have created a critical need to continuously learn and discover new material phases or defect types. Therefore, C-GCD is positioned to unlock greater potential in AI-powered scientific discovery across various domains. In this work, we focus primarily on addressing the C-GCD problem in the context of general natural images and do not design specialized mechanisms for more fine-grained scenarios. We will focus on these scenarios in the future

Table 5. 25-Stage Performance on Tiny-ImageNet.

Methods	Stage	1	2	3	4	5	6	7	8	9	10
VanillaGCD	All	79.44	77.35	75.86	73.90	72.70	70.85	70.25	69.11	66.29	64.67
	Old	81.34	78.81	77.76	74.82	73.91	72.50	71.50	70.13	67.55	66.03
	New	32.00	39.50	24.50	48.00	37.50	21.50	31.50	36.50	25.00	18.50
Happy	All	83.04	80.70	79.91	78.38	76.70	75.79	73.30	71.71	67.87	65.23
	Old	83.24	81.50	79.35	77.95	76.19	77.28	72.85	72.78	69.15	67.12
	New	78.00	60.00	95.00	90.50	91.50	31.00	87.00	37.50	25.50	1.00
DYDM (Ours)	All	83.33	82.44	82.00	81.00	80.90	79.82	79.16	78.02	76.69	75.50
	Old	83.96	82.81	82.00	81.77	80.78	80.52	79.53	78.77	77.64	76.41
	New	67.50	73.00	82.00	59.50	84.50	59.00	67.50	54.00	45.50	44.50

Methods	Stage	11	12	13	14	15	16	17	18	19	20
VanillaGCD	All	62.39	61.65	60.46	58.14	57.41	56.01	54.54	54.60	53.02	52.06
	Old	63.59	62.94	61.78	59.21	58.19	56.91	55.23	55.51	53.98	52.66
	New	20.50	15.00	11.50	17.50	27.00	20.00	26.00	16.50	12.00	25.50
Happy	All	62.61	60.18	58.92	57.62	54.66	53.61	55.38	53.45	48.77	49.24
	Old	63.79	61.25	60.15	59.11	54.26	54.95	56.09	54.17	49.66	49.89
	New	21.50	21.50	13.50	1.00	70.50	0.00	26.50	23.50	10.50	21.00
DYDM (Ours)	All	75.58	74.50	72.97	72.19	71.61	70.93	69.79	69.06	68.39	67.70
	Old	75.46	75.40	74.03	72.57	72.04	71.50	70.43	69.52	68.63	68.00
	New	80.00	42.00	34.00	58.00	55.00	48.00	43.50	49.50	58.00	54.50

Methods	Stage	21	22	23	24	25	Avg ACC
VanillaGCD	All	51.32	49.17	49.30	47.95	46.64	61.40
	Old	51.82	49.93	50.02	48.25	47.02	62.46
	New	28.50	14.00	15.50	33.50	28.00	25.02
Happy	All	45.91	45.41	44.94	42.54	42.56	61.14
	Old	46.50	45.86	45.88	42.41	42.92	61.77
	New	19.50	25.00	0.50	49.00	25.00	37.02
DYDM (Ours)	All	67.33	66.81	66.39	66.01	65.05	**73.73**
	Old	67.54	67.15	66.56	66.08	65.74	**74.19**
	New	57.50	51.00	58.00	62.50	31.00	**56.78**