Semantic Grouping with Dual-Strategy Distributional Rehearsal for Continual Learning

Nho-Khoa Le [1] **Xuan-Hung Ho** [2] **Thanh-Hai Dang** [1] **Duc-Trong Le** [1] **Quynh-Trang Pham Thi** [1]

Abstract

Continual learning systems face catastrophic forgetting, where acquiring new knowledge degrades performance on prior tasks. We propose **SGC**, a novel exemplar-free framework addressing this through three key innovations: (1) a Max-Cut graph-based partitioning strategy that groups incoming classes into semantically distinct groups; (2) a dual-strategy distributional rehearsal (DSDR) combining archetype interpolation and GMM sampling to generate high-fidelity synthetic features preserving past class distributions; and (3) a group-focused training regimen that updates only lightweight group-specific modules while freezing the shared backbone. Extensive experiments on CIFAR-100 and four fine-grained datasets (CUB-200, Flower, Stanford-Cars, Food-101) demonstrate superior performance over state-of-the-art exemplar-free methods.

1. Introduction

Deep learning suffers from catastrophic forgetting in sequential learning (Wang et al., 2024), where weight updates for new tasks erase prior knowledge, which is a critical barrier for real-world applications. Continual Learning (CL) addresses this through replay-based (Ho et al., 2023), regularization-based (Kirkpatrick et al., 2017), and Bayesian methods (Li et al., 2025), each facing trade-offs between memory, computational cost, and plasticity (Qiu et al., 2024). A key gap remains: existing methods lack a framework that is simultaneously scalable, exemplar-free, and semantically aware of incoming data. Most treat new classes as a monolithic block and rely on oversimplified distributional assumptions, leading to a suboptimal stability-plasticity balance. To address this, we propose a CL framework that uses Max-Cut optimization to partition incoming classes into semantically distinct groups, trains lightweight group-specific modules within a dynamic modular architecture, and introduces a Dual-Strategy Distributional Rehearsal (DSDR) mechanism combining Gaussian Mixture Models (GMMs) and archetype-based interpolation for high-fidelity exemplar-free rehearsal. Our primary contributions are:

- A Max-Cut-based class grouping strategy that partitions incoming classes into semantically distinct groups.
- A DSDR mechanism combining GMM sampling and archetype interpolation for exemplar-free rehearsal.
- A group-focused training regimen with a frozen backbone and lightweight group-specific modules for effective stability-plasticity balance.
- State-of-the-art performance among exemplar-free methods on CIFAR-100 and four fine-grained datasets.

2. Methodology

In this section, we present our novel framework for continual learning, designed to mitigate catastrophic forgetting by dynamically organizing incoming classes into semantically distinct groups and employing a sophisticated pseudo-rehearsal strategy. Our core idea is to move beyond class-agnostic replay and instead create specialized, interpretable modules for distinct class groups. This is achieved through a three-stage pipeline for each incremental task: 1) a graph-based partitioning of new classes to form semantic groups, 2) a rich, dual-strategy modeling of class feature distributions to enable high-fidelity pseudo-rehearsal, and 3) a group-focused training regimen that updates modular components of our dynamic modular architecture.

2.1. Problem Formulation

We address the class-incremental learning scenario, where the task id is not provided in the testing phase. The model is presented with a sequence of tasks $T = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$, where each task $\mathcal{T}_t$ introduces a dataset $\mathcal{D}_t = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_t}$

[1]Vietnam National University, University of Engineering and Technology, Hanoi, Vietnam [2]Viettel Networks, Viettel Group. Correspondence to: Quynh-Trang Pham Thi <trangptq@vnu.edu.vn>.

Presented at the ICML 2026 Workshop "Continual Adaptation at Scale: Towards Sustainable AI". Copyright 2026 by the author(s).

that contains samples from a set of new classes $\mathcal{C}_t$. The sets of classes are disjoint, that is, $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ for $i \neq j$. After training on task $\mathcal{T}_t$, the model must accurately classify samples from all previously seen classes, $\mathcal{C}_{1:t} = \bigcup_{i=1}^{t} \mathcal{C}_i$. A critical constraint is that the data from previous tasks, $\mathcal{D}_{1:t-1}$, is not accessible during training on $\mathcal{T}_t$.

2.2. Framework Overview

Our proposed method, depicted in Figure 1, processes each new task $\mathcal{T}_t$ through a structured pipeline. Upon the arrival of new classes $\mathcal{C}_t$, we first model their feature distributions using a frozen backbone network. This involves computing not only a mean prototype but also a more descriptive representation using K-Means to find class *archetypes* and fitting a Gaussian Mixture Model (GMM) to capture intra-class variance.

Concurrently, we perform Incremental Class Grouping by constructing a similarity graph over the new classes and partitioning it using a Max-Cut algorithm. This separates the classes into distinct, semantically dissimilar groups. For each new group formed, our model architecture dynamically expands by adding a dedicated *fully connected network* and a *classification head*.

To combat forgetting, we introduce a Balanced Pseudo-Rehearsal strategy. We generate synthetic features for all past classes ($\mathcal{C}_{1:t-1}$) by combining two techniques: sampling from their stored GMM parameters and interpolating between their learned archetypes. Finally, we initiate a Group-Focused Training stage, where only the classification heads and explainers are trained on a dataset comprising real features for new classes and synthetic features for old ones. The shared feature extractor remains frozen throughout all incremental steps after the initial task, ensuring knowledge stability.

2.3. Incremental Class Grouping via Max-Cut

A key innovation of our approach is the dynamic organization of classes into manageable groups. This simplifies the classification problem into a set of smaller, more specialized sub-problems. This process operates as follows:

1. **Prototype Extraction**: For each new class $k \in \mathcal{C}_t$, we compute its mean feature vector, or prototype, $\mathbf{c}_k$, by averaging the embeddings of its training samples produced by the frozen backbone.

2. **Graph Construction**: We construct a fully connected, undirected SimGraph $G = (\mathcal{V}, \mathcal{E})$, where the set of vertices $\mathcal{V}$ corresponds to the new classes $\mathcal{C}_t$. The weight w_{ij} of the edge between two vertices (classes) i and j is defined by the cosine similarity of their prototypes:

$$w_{ij} = \frac{\mathbf{c}_i \cdot \mathbf{c}_j}{\|\mathbf{c}_i\| \|\mathbf{c}_j\|} \tag{1}$$

   High similarity indicates that two classes are close in the feature space and are not candidates for being grouped together.

3. **Graph Partitioning**: We formulate the grouping task as a **Max-Cut** problem. The objective is to partition the vertices $\mathcal{V}$ into two disjoint sets, S_1 and S_2, such that the sum of weights of edges connecting a vertex in S_1 to a vertex in S_2 is maximized. This partitions the classes into two maximally dissimilar super-classes or groups. To handle tasks with a large number of incoming classes ($|\mathcal{C}_t| > 20$), we employ an iterative strategy, applying the Max-Cut partitioning on smaller, computationally tractable chunks of classes.

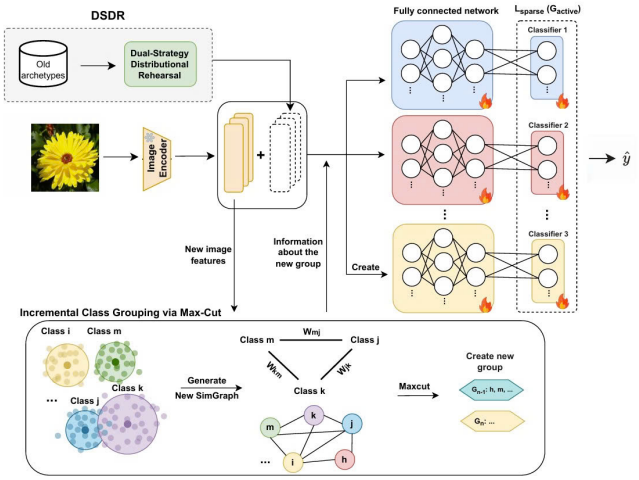


Figure 1. Overview of our proposed framework for a single incremental task. New data is used to model class distributions and partition classes into groups. The model then expands, and it is trained using real data for new classes and generated pseudo-features for old classes.

2.4. Dual-Strategy Distributional Rehearsal

To effectively prevent catastrophic forgetting, a replay mechanism must generate features that are not only realistic but also diverse. Relying on a single class prototype is insufficient. We therefore model the feature distribution of each class using a dual strategy. For each previously seen class $k \in \mathcal{C}_{1:t-1}$, we store its distributional parameters.

1. **Archetype-based Interpolation**: We first identify M representative *archetypes* $\{\mathbf{a}_{k,m}\}_{m=1}^{M}$ for each class k by applying K-Means clustering to its feature representations during its initial training. These archetypes

capture the primary modes of the class manifold. Synthetic samples are then generated by random convex combinations of these archetypes:

$$\tilde{\mathbf{x}}_{\text{interp}} = \sum_{m=1}^{M} \alpha_m \mathbf{a}_{k,m}, \quad \text{s.t.} \sum_{m=1}^{M} \alpha_m = 1, \alpha_m \geq 0 \tag{2}$$

This approach generates samples that lie on the convex hull of the class's primary modes, preserving the global structure of the feature distribution.

2. **Gaussian Mixture Model Sampling**: To capture local density and variation, we also model the feature distribution of each archetype cluster as a multivariate Gaussian. This results in a Gaussian Mixture Model (GMM) for each class, with parameters $\{(\pi_{k,m}, \boldsymbol{\mu}_{k,m}, \boldsymbol{\Sigma}_{k,m})\}_{m=1}^{M}$, where $\pi_{k,m}$ denotes the mixture weight of the m-th Gaussian component for class k, $\boldsymbol{\mu}_{k,m}$ is its mean vector, and $\boldsymbol{\Sigma}_{k,m}$ is the covariance matrix. We generate a second set of synthetic samples by sampling from this learned GMM.

The final set of pseudo-features is a balanced mixture of samples from both strategies, controlled by a hyperparameter ρ that dictates the interpolation ratio. This dual approach ensures that our replay mechanism generates features that represent both the structural manifold and the local density of the past data.

2.5. Group-Focused Modular Training

Our network architecture starts with a shared feature encoder, f_{enc}, which processes an input image $\mathbf{x}$ into a feature vector $\mathbf{z} = f_{enc}(\mathbf{x})$. This is followed by a dynamically growing set of group-specific modules. For each group g, we maintain an *fully connected network* f_g^{fc}, and a linear *classification head*, f_g^{head}.

During inference, the feature vector $\mathbf{z}$ is fed into each available fully connected network. Each fully connected network f_g^{fc} independently generates a group-specific activation vector $\mathbf{p}_g = f_g^{\text{fc}}(\mathbf{z})$. This vector is then immediately passed to its corresponding classification head f_g^{head}, which produces logits only for the classes within that group. The final output is a combined logits vector, formed by arranging the group-specific logits to represent all previously seen classes.

$$z = f_{\text{enc}}(x), \tag{3}$$

$$p_g = f_g^{\text{fc}}(z), \quad \forall g \in \mathcal{G}, \tag{4}$$

$$\ell = \text{Concat}\big(\{f_g^{\text{head}}(p_g) \mid g \in \mathcal{G}\}\big). \tag{5}$$

The training process is carefully constrained to ensure knowledge preservation and effective learning of new tasks.

Table 1. Performance comparison on CIFAR-100. The best performance is shown in bold, and the second-best performance is underlined.

Methods	B-10 Inc-10		B-50 Inc-5	
	$\bar{A}$	A_{last}	$\bar{A}$	A_{last}
L2P(Wang et al., 2022b)	76.56	65.75	61.85	44.19
DualPrompt(Wang et al., 2022a)	81.41	70.34	64.05	43.86
CODA-Prompt(Smith et al., 2023)	82.13	72.34	65.49	49.72
CPP(Li et al., 2024)	75.73	67.50	69.57	66.26
LAE(Gao et al., 2023)	82.65	72.60	67.35	50.96
Continual-CLIP(Thengane et al., 2022)	75.15	66.68	70.79	66.68
SLCA(Zhang et al., 2023)	83.13	72.01	<u>80.07</u>	71.24
EASE(Zhou et al., 2024)	<u>85.07</u>	<u>77.31</u>	76.72	70.50
CLG-CBM(Yu et al., 2025)	84.49	76.82	79.07	<u>75.91</u>
SGC(Ours)	**87.36**$_{\pm 0.45}$	**80.95**$_{\pm 0.14}$	**83.81**$_{\pm 0.33}$	**80.92**$_{\pm 0.11}$

1. **Parameter Isolation**: During an incremental step, the backbone f_{enc} is kept frozen. Only the parameters of the classification heads and fully connected networks for all groups (both old and new) are updated.

2. **Loss Function**: The trainable parameters are optimized using a composite loss function. For each batch $\mathcal{B}$, the total loss $\mathcal{L}_{\mathcal{B}}$ is a weighted sum of the standard classification loss and a sparsity-inducing regularizer that is **dynamically applied only for the groups active in that batch**:

$$\mathcal{L}_{\mathcal{B}} = \mathcal{L}_{CE}(\hat{y}, y) + \lambda \sum_{g \in \mathcal{G}_{\mathcal{B}}} \|W_g\|_1 \tag{6}$$

where $\mathcal{L}_{CE}$ is the Cross-Entropy loss, W_g are the weights of the classification head for group g, and $\mathcal{G}_{\mathcal{B}}$ is the set of unique group indices corresponding to the class labels present in the current batch $\mathcal{B}$. The key insight behind this formulation is twofold. Primarily, by localizing the sparsity pressure to only the active groups within a batch, we shield the group-specific parameters of inactive groups from unnecessary updates. This targeted approach acts as a form of implicit regularization masking, directly contributing to knowledge preservation and mitigating catastrophic forgetting. Secondarily, this approach is also more computationally efficient. The coefficient λ controls the strength of this sparsity constraint.

3. Experimental Results

3.1. Experiments Setup

Evaluation Benchmarks. We evaluate our method on a comprehensive datasets: CIFAR-100 (Krizhevsky et al., 2009), CUB-200 (Wah et al., 2011), Flower (Nilsback & Zisserman, 2008), Food-101 (Bossard et al., 2014), and Stanford-Cars (Krause et al., 2013). We evaluate on two primary metrics: *average incremental accuracy*($\bar{A}$) and *final average accuracy*(A_{last})

Implementation Details. Across all experiments, we employed CLIP ViT-B/16 as the default feature extractor for

Table 2. Average incremental accuracy comparison on four datasets, the best performance is shown in bold, the second-best is underlined. All methods are implemented without using exemplars. We replace the backbones of all methods with CLIP ViT-B/16.

Methods	CUB-200		Flower		Stanford-Cars		Food-101	
	B-10 Inc-10	B-100 Inc-10	B-10 Inc-10	B-50 Inc-5	B-14 Inc-14	B-100 Inc-10	B-10 Inc-10	B-50 Inc-5
L2P(Wang et al., 2022b)	62.08	59.38	84.37	76.70	64.42	61.82	79.48	69.72
DualPrompt(Wang et al., 2022a)	64.95	61.85	89.64	79.16	76.94	68.46	86.27	69.66
CODA-Prompt(Smith et al., 2023)	67.22	59.82	88.57	77.70	76.44	60.80	87.76	67.22
CPP(Li et al., 2024)	83.60	75.03	94.95	93.30	84.75	77.49	90.21	86.60
LAE(Gao et al., 2023)	66.45	59.98	86.79	77.55	77.25	80.28	88.41	66.26
Continual-CLIP(Thengane et al., 2022)	69.41	60.35	78.72	74.06	86.43	69.79	92.04	89.96
SLCA(Zhang et al., 2023)	80.53	76.85	92.77	82.14	84.74	70.59	74.49	76.28
EASE(Zhou et al., 2024)	83.87	66.14	94.86	77.05	86.22	64.32	91.74	81.72
CLG-CBM(Yu et al., 2025)	<u>85.40</u>	<u>82.20</u>	<u>95.58</u>	<u>94.53</u>	<u>88.60</u>	<u>85.07</u>	<u>92.25</u>	<u>90.97</u>
SGC(Ours)	**86.18**$_{\pm 0.41}$	**82.59**$_{\pm 0.21}$	**96.36**$_{\pm 0.22}$	**95.67**$_{\pm 0.32}$	**90.36**$_{\pm 0.38}$	**87.26**$_{\pm 0.28}$	**94.08**$_{\pm 0.06}$	**92.63**$_{\pm 0.12}$

all models, unless specified differently. We used the Adam optimizer (Kingma, 2014) with a learning rate of 0.001 and a batch size of 64 for a total of 60 epochs. Furthermore, the weight of the sparsity loss, λ to 0.001 and the interpolation ratio, ρ was set at 0.3.

3.2. Result and Discussion

Evaluation on CIFAR-100 Dataset. As shown in Table 1, SGC sets a new state-of-the-art on the CIFAR-100 benchmark. In the B-10 Inc-10 setting, we achieve an average incremental accuracy ($\bar{A}$) of 87.36% and a final average accuracy (A_{last}) of 80.95%, outperforming the previous best method by 2.29% and 3.64%, respectively. The dominance of SGC is even more pronounced in the B-50 Inc-5 setting, where the SGC model achieves 83. 81% in $\bar{A}$ and 80. 92% in A_{last}, surpassing the strongest competitors by at least 3.74% and 5.01%. These results clearly demonstrate the superior performance of our approach.

Evaluation on Fine-grained Datasets. We further evaluate SGC on 4 fine-grained datasets: CUB-200, Flower, Food-101 and Standford-cars. As shown in Table 2, SGC achieves the best results on all four datasets. Compared with the strongest baseline (CLG-CBM), it improves by 0.78%/0.39% on CUB-200, 0.78%/1.14% on Flowers, 1.76%/2.19% on Stanford-Cars, and 1.83%/1.66% on Food-101. These consistent improvements highlight the robustness and generalizability of our approach across diverse benchmarks.

4. Conclusion

In this work, we propose SGC, a novel continual learning framework that effectively addresses catastrophic forgetting through semantically-aware class organization and sophisticated pseudo-rehearsal mechanisms. Our approach introduces three key innovations: a graph-based Max-Cut partitioning strategy for dynamic class grouping, a dual-strategy distributional rehearsal mechanism combining GMM-based sampling with archetype-based interpolation, and a group-focused training regimen that maintains knowledge stability while enabling efficient adaptation. Extensive experiments on CIFAR-100 and four challenging fine-grained datasets demonstrate that our method consistently outperforms state-of-the-art exemplar-free approaches, achieving superior average incremental accuracy and significantly reduced forgetting scores. The proposed framework's ability to operate effectively without storing previous task data while maintaining high performance across diverse datasets highlights its practical value for real-world continual learning applications. Future work could explore extending this approach to more complex continual learning scenarios and investigating the scalability of the semantic grouping strategy to larger numbers of tasks and classes.

Acknowledgements

This research was partially supported by AI Viet Nam.

References

Bossard, L., Guillaumin, M., and Van Gool, L. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

Douillard, A., Cord, M., Ollion, C., Robert, T., and Valle, E. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *European conference on computer vision*, pp. 86–102. Springer, 2020.

Fortunato, S. and Hric, D. Community detection in networks: A user guide. *Physics reports*, 659:1–44, 2016.

Gao, Q., Zhao, C., Sun, Y., Xi, T., Zhang, G., Ghanem, B., and Zhang, J. A unified continual learning framework with general parameter-efficient tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11483–11493, 2023.

Ghashami, M., Liberty, E., Phillips, J. M., and Woodruff, D. P. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.

Goemans, M. X. and Williamson, D. P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

Hamilton, W. L., Ying, R., and Leskovec, J. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

Ho, S., Liu, M., Du, L., Gao, L., and Xiang, Y. Prototype-guided memory replay for continual learning. *IEEE transactions on neural networks and learning systems*, 35(8): 10973–10983, 2023.

Hou, S., Pan, X., Loy, C. C., Wang, Z., and Lin, D. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 831–839, 2019.

Javed, K. and White, M. *Meta-learning representations for continual learning*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Kingma, D. P. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*. International Conference on Learning Representations, 2014.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Li, M., Lu, Y., Dai, Q., Huang, S., Ding, Y., and Lu, H. Became: Bayesian continual learning with adaptive model merging. In *Forty-second International Conference on Machine Learning*, 2025.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Li, Z., Zhao, L., Zhang, Z., Zhang, H., Liu, D., Liu, T., and Metaxas, D. N. Steering prototypes with prompt-tuning for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2523–2533, 2024.

Newman, M. E. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

Nilsback, M.-E. and Zisserman, A. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Qiu, Z., Xu, Y., Meng, F., Li, H., Xu, L., and Wu, Q. Dual-consistency model inversion for non-exemplar class incremental learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24025–24035. IEEE Computer Society, 2024.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Smith, J. S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., and Kira, Z. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11909–11919, 2023.

Thengane, V., Khan, S., Hayat, M., and Khan, F. Clip model is an efficient continual learner. *arXiv preprint arXiv:2210.03114*, 2022.

Von Luxburg, U. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

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

Wang, L., Zhang, X., Su, H., and Zhu, J. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5362–5383, 2024.

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.-Y., Ren, X., Su, G., Perot, V., Dy, J., et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pp. 631–648. Springer, 2022a.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., and Pfister, T. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 139–149, 2022b.

Yoon, J., Lee, J., Yang, E., and Hwang, S. J. Lifelong learning with dynamically expandable network. In *International Conference on Learning Representations*. International Conference on Learning Representations, 2018.

Yu, L., Han, H., Tao, Z., Yao, H., and Xu, C. Language guided concept bottleneck models for interpretable continual learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14976–14986, 2025.

Zhang, G., Wang, L., Kang, G., Chen, L., and Wei, Y. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19148–19158, 2023.

Zhou, D.-W., Sun, H.-L., Ye, H.-J., and Zhan, D.-C. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23554–23564, 2024.

Zhu, X. J. Semi-supervised learning literature survey. 2005.

A. Appendix

A.1. Class Incremental Learning

Class incremental learning is a scenario in continual learning in which new classes arrive sequentially without explicit task boundaries. Unlike task-incremental learning, models must distinguish between all previously seen classes while incorporating new ones, making the problem significantly more complex due to the absence of task identifiers during inference. Early approaches to class incremental learning focused on knowledge distillation and architectural modifications. Learning without Forgetting (LwF) (Li & Hoiem, 2017) employs knowledge distillation to preserve outputs on old classes while learning new ones. iCaRL (Rebuffi et al., 2017) combines exemplar storage with nearest-mean classification, storing representative samples from previous classes. LUCIR (Hou et al., 2019) addresses the bias towards new classes through cosine normalization and margin ranking loss, while PODNet (Douillard et al., 2020) uses spatial-based distillation to preserve intermediate representations.

Recent advances have explored prompt-based approaches that achieve impressive performance without storing exemplars. Learning to Prompt (L2P) (Wang et al., 2022b) learns a pool of prompts and selectively uses them for different tasks, while DualPrompt (Wang et al., 2022a) employs complementary prompts for general and specific knowledge. CODA-Prompt (Smith et al., 2023) further improves this paradigm through continual decomposed attention-based prompting. These methods demonstrate that frozen pre-trained models can be effectively adapted for continual learning through careful prompt design.

Parameter-efficient approaches have also gained attention in class incremental learning. CPP (Li et al., 2024) steers prototypes with prompt-tuning for rehearsal-free continual learning, while LAE (Gao et al., 2023) provides a unified framework with general parameter-efficient tuning. SLCA (Zhang et al., 2023) focuses on slow learning with classifier alignment, and EASE (Zhou et al., 2024) proposes expandable subspace ensembles for pre-trained model-based learning.

Despite these advances, most existing methods treat incoming classes as independent entities, failing to take advantage of potential semantic relationships that could simplify the learning problem. This limitation becomes particularly pronounced when dealing with fine-grained datasets where classes may share substantial semantic overlap. Our approach addresses this gap by dynamically organizing classes into semantically distinct groups, enabling more efficient and targeted learning strategies.

A.2. Graph-Driven Dynamic Similarity Grouping

Graph-based methods have demonstrated significant success in capturing and exploiting similarity relationships across various machine learning domains. In clustering and community detection, spectral methods (Von Luxburg, 2007) and modularity optimization (Newman, 2006) leverage graph structures to identify meaningful groupings. Graph partitioning algorithms, particularly the Max-Cut problem (Goemans & Williamson, 1995), have been extensively studied for dividing vertices into disjoint sets that maximize inter-group connections.

In the context of machine learning, graph-based similarity grouping has been applied to semi-supervised learning (Zhu, 2005), where label propagation on similarity graphs enables learning from limited labeled data. Graph neural networks (Hamilton et al., 2017) have further advanced the field by learning representations that incorporate graph structure, enabling more sophisticated similarity-based reasoning.

Dynamic grouping approaches have emerged to handle evolving data distributions and relationships. Online clustering methods (Ghashami et al., 2016) adapt to streaming data by continuously updating cluster assignments, while dynamic community detection algorithms (Fortunato & Hric, 2016) track evolving community structures in temporal networks. These methods typically focus on maintaining consistency while adapting to new information.

In continual learning, some works have explored graph structures for organizing knowledge, though primarily at the task level rather than the class level. Meta-learning approaches (Javed & White, 2019) use task similarity graphs to guide parameter sharing, while some architectural methods (Yoon et al., 2018) employ graph structures to determine parameter allocation across tasks.

However, existing graph-based approaches in continual learning have limitations in addressing class-level semantic organization within individual tasks. Most methods either operate at the task level or require static, pre-defined groupings. Our approach introduces a novel application of graph partitioning specifically for dynamic class grouping, where we construct similarity graphs from class prototypes and employ Max-Cut optimization to create semantically distinct clusters.

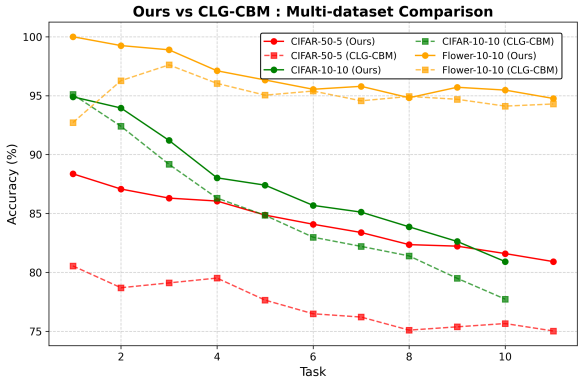


Figure 2. Comparison of accuracy across tasks for different datasets (CIFAR-50-5, CIFAR-10-10, and Flower-10-10) using our proposed method (solid lines) and the CLG-CBM baseline (dashed lines). The results show that our approach consistently achieves higher accuracy than CLG-CBM, with the performance gap being more pronounced on CIFAR datasets, while both methods maintain high accuracy on Flower-10-10.

This enables the decomposition of complex classification problems into smaller, more manageable sub-problems, while maintaining the flexibility to adapt to new class configurations as they arrive.

A.3. Further Result Analysis.

To further validate the effectiveness of SGC, we performed a detailed comparative analysis against the baseline CLG-CBM on various challenging continual learning benchmarks. Figure 2 compares the accuracy curves of our method and CLG-CBM across multiple datasets. Our approach consistently maintains higher accuracy throughout the incremental tasks, with a notably larger margin on CIFAR-50-5 and CIFAR-10-10. On Flower-10-10, both methods achieve relatively stable performance; SGC shows a slight advantage, confirming its effectiveness across diverse domains. Figure 3 further illustrates

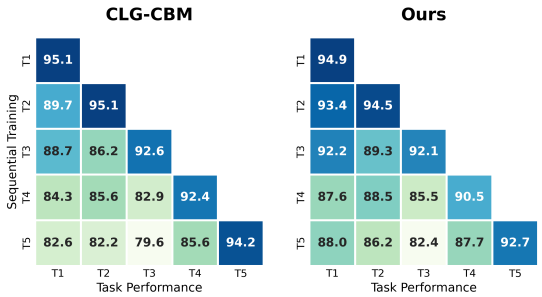


Figure 3. Accuracy progression across sequential tasks on CIFAR B-10 Inc-10. The left heatmap shows the performance of the CLG-CBM model, while the right heatmap illustrates our method. Each cell indicates the accuracy of a model trained up to task i when evaluated on task j ($i \leq j$).

the progression of accuracy on CIFAR B-10 Inc-10. The heatmaps reveal that our method preserves knowledge from earlier tasks more effectively than CLG-CBM, resulting in consistently higher accuracy on past tasks. Although our model sometimes shows slightly lower performance when learning new tasks, its ability to retain previous knowledge is markedly superior. This observation is further supported by the forgetting score results in Table 3, where our approach achieves a significantly lower score (10.99%) compared to CLG-CBM (14.32%). This improvement highlights the effectiveness of our rehearsal-based DSDR strategy, which plays a crucial role in mitigating catastrophic forgetting and maintaining stable performance across tasks.

Method	Forgetting Score (%)
CLG-CBM	14.32
SGC(Ours)	**10.99**

Table 3. Comparison of forgetting scores between SGC and CLG-CBM on CIFAR B-10 Inc-10. Note that lower forgetting scores indicate better performance.

To better understand the contribution of each component, we conduct an ablation study as shown in Table 4. The results demonstrate that removing DSDR and reverting to SGPA, Semantic Guide Prototype Augmentation (CLG-CBM), or discarding grouping, both lead to performance drops across datasets. Our complete model consistently achieves the best results, validating both the effectiveness of DSDR and the grouping strategy.

Method	**CIFAR100** B-10 Inc-10	**Food-101** B-10 Inc-10	**Stanford-Cars** B-14 Inc-14
w/o DSDR (SGPA)	81.46	93.41	89.95
w/o grouping	83.50	93.66	89.73
SGC (Ours)	**83.81**	**94.08**	**90.36**

Table 4. Ablation study on SGC components across datasets.

A.4. Additional Memory and Baseline Analysis

We further analyze whether the performance of SGC is mainly due to a larger memory footprint. Although SGC does not store raw images, it stores distributional statistics for pseudo-feature generation. The dominant term is the covariance matrix of CLIP features. With CLIP ViT-B/16, each feature has dimension $D = 512$, and a full fp32 covariance requires

$$D^2 \times 4 = 512^2 \times 4 = 1{,}048{,}576 \text{ bytes}, \tag{7}$$

which is approximately 1 MiB per covariance matrix.

To reduce this cost, we evaluate a low-rank plus diagonal covariance representation:

$$\mathbf{\Sigma} \approx \mathbf{F}\mathbf{F}^\top + \mathrm{diag}(\mathbf{d}), \tag{8}$$

where $\mathbf{F} \in \mathbb{R}^{512\times r}$ stores the leading covariance directions and $\mathbf{d} \in \mathbb{R}^{512}$ stores the residual diagonal variance. We use $r = 128$ and fp16 storage. Thus, one covariance requires

$$512 \times 128 \times 2 + 512 \times 2 = 132{,}096 \text{ bytes}, \tag{9}$$

which is about $7.9\times$ smaller than the original fp32 covariance. During rehearsal, pseudo-features are sampled as

$$\tilde{\mathbf{z}} = \boldsymbol{\mu} + \boldsymbol{\epsilon}_r \mathbf{F}^\top + \boldsymbol{\epsilon}_d \odot \sqrt{\mathbf{d}}, \tag{10}$$

where $\boldsymbol{\epsilon}_r \sim \mathcal{N}(0, I_r)$ and $\boldsymbol{\epsilon}_d \sim \mathcal{N}(0, I_{512})$.

Table 5. Memory-light SGC results. Full-covariance numbers are the main SGC results reported in the paper, while memory is estimated from the stored SGC state at the final task.

Dataset	**SGC variant**	$\bar{A}$	A_{last}	**Memory**
CIFAR-100	Full covariance fp32	87.36	80.95	502.93 MiB
CIFAR-100	Low-rank+diag fp16, $r = 128$	87.35	80.80	66.79 MiB
CUB-200	Full covariance fp32	86.18	82.59	743.96 MiB
CUB-200	Low-rank+diag fp16, $r = 128$	86.62	79.65	99.80 MiB

As shown in Table 5, the low-rank variant reduces the overall SGC state by approximately $7.5\times$ on both CIFAR-100 and CUB-200. On CIFAR-100, the accuracy is almost unchanged. On CUB-200, the average accuracy remains competitive, while the final accuracy decreases, suggesting that the compact representation preserves the dominant distributional geometry but may lose some fine-grained tail information.

Table 6. Comparison with memory-matched raw feature replay. The replay baseline uses a similar byte budget to store old CLIP feature vectors directly.

Dataset	Method	$\bar{A}$	A_{last}	Memory
CIFAR-100	Raw feature replay	84.17	79.05	67.04 MiB
CIFAR-100	SGC low-rank covariance	**87.35**	**80.80**	66.79 MiB
CUB-200	Raw feature replay	74.76	74.73	108.06 MiB
CUB-200	SGC low-rank covariance	**86.62**	**79.65**	99.80 MiB

Table 6 shows that SGC remains stronger than raw feature replay under a comparable memory budget. This is particularly informative on CUB-200, where the replay budget is large enough to store almost all available training features due to the small number of images per class. Nevertheless, replay remains below SGC, indicating that the advantage is not simply from storing more old information. Instead, SGC benefits from modeling class distributions and generating pseudo-features through the dual-rehearsal strategy.

Table 7. Additional frozen-CLIP baselines. Joint linear probing is an offline reference and is not a fair continual-learning baseline.

Dataset	Method	$\bar{A}$	A_{last}
CIFAR-100	CLIP zero-shot	78.23	67.53
CIFAR-100	Nearest-mean CLIP	79.05	70.00
CIFAR-100	Joint linear probe	–	**82.52**
CIFAR-100	SGC low-rank covariance	**87.35**	80.80
CUB-200	CLIP zero-shot	69.00	55.99
CUB-200	Nearest-mean CLIP	85.35	77.05
CUB-200	Joint linear probe	–	**81.41**
CUB-200	SGC low-rank covariance	**86.62**	79.65

The additional baselines in Table 7 provide useful context. Nearest-mean CLIP is especially strong on CUB-200 because it uses real training image embeddings to build class prototypes, and fine-grained bird classes are well clustered in the frozen CLIP image feature space. In contrast, CLIP zero-shot relies only on class-name prompts and does not use any training images. For fine-grained bird species, class names alone may not encode the subtle visual distinctions required for accurate classification, which explains the large gap between nearest-mean and zero-shot on CUB-200. Finally, the joint linear probe uses all training features at once and should be interpreted only as an offline reference rather than a continual-learning baseline.