DyCoG: Dynamic Concept Pool with Instance-Adaptive Gating for Interpretable Continual Learning

Anonymous authors
Paper under double-blind review

Abstract

Continual learning (CL) aims to acquire new knowledge from evolving tasks while preserving previously learned knowledge. Interpretable CL has attracted increasing attention for its potential to achieve strong continual learning performance while providing more transparent decision processes. However, existing methods often construct semantic concepts only once per incremental stage, limiting their adaptability to evolving class relationships, while final predictions still rely on trainable classification modules that may accumulate optimization errors over time. To address these limitations, we propose **Dynamic Concept Pool with Instance-Adaptive Gating (DyCoG)**, an interpretable continual learning framework that grounds both knowledge accumulation and classification in explicit semantic concepts. Specifically, the **Dynamic Concept Pool (DCP)** progressively refines a human-understandable class-level concept space through classification feedback as new classes are introduced, enabling adaptive and interpretable knowledge accumulation. The **Instance-Adaptive Concept Gate (IACG)** further selects an instance-specific subset of concepts for each input, making the final decision directly traceable to instance-relevant semantic evidence. The resulting concept representations are directly used for statistical classification without relying on trainable classification modules for final prediction. Experiments on four benchmark datasets demonstrate that DyCoG delivers strong continual learning performance with competitive computational cost. More notably, as task partitions become finer, DyCoG exhibits an overall performance improvement rather than degradation, in contrast to the declining or saturating trends observed in most compared methods.

1 Introduction

Continual learning (CL) aims to enable models to continuously acquire new knowledge from evolving tasks while retaining previously learned knowledge (Wang et al., 2024; Elsayed & Mahmood, 2024; Masana et al., 2023). Compared with conventional deep learning, where models are trained once on a fixed dataset, continual learning allows models to be continuously updated as new tasks arrive without repeatedly retraining from scratch, making it applicable to dynamic environments. However, during continual updates, learning new knowledge may interfere with previously acquired representations, leading to substantial performance degradation on earlier tasks or classes, a phenomenon known as catastrophic forgetting (McCloskey & Cohen, 1989; Davari et al., 2022).

To mitigate catastrophic forgetting, existing continual learning methods mainly adopt strategies such as knowledge replay (Lin et al., 2023; Tong et al., 2025), parameter regularization (Li et al., 2024; Smith et al., 2023b; Zhao et al., 2024), and architectural adaptation (Lanzillotta et al., 2023; Douillard et al., 2022; Lu et al., 2025). Although these methods can alleviate forgetting to some extent, they mainly rely on training constraints or architectural modifications, paying relatively limited attention to feature representation and decision mechanisms. In recent years, interpretable CL has attracted increasing attention and emerged as an active research direction, benefiting from its ability to provide semantic representations and more transparent decision processes (Lorello et al., 2024; Lai et al., 2025b; Di Valerio et al., 2026).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

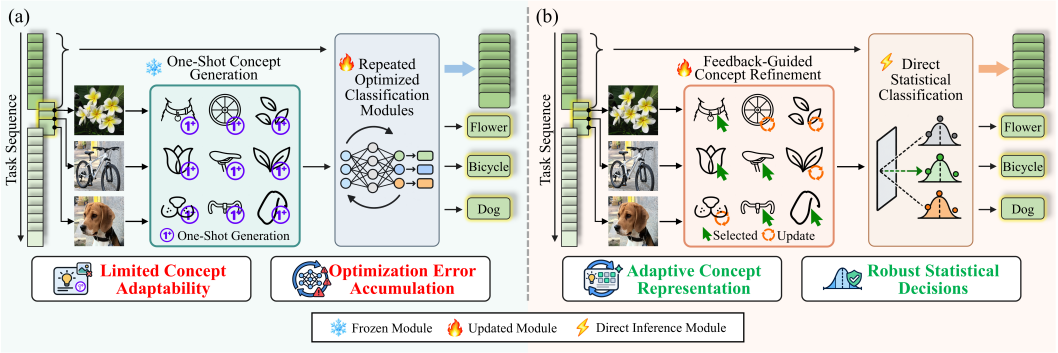

Figure 1: Motivation of DyCoG. (a) Existing interpretable CL methods generate concepts once from class names and rely on trainable classification modules, which leads to limited concept adaptability and accumulated optimization errors. (b) DyCoG instead performs feedback-guided concept refinement and directly classifies from concept statistics, yielding adaptive concept representations and more stable decisions.

Existing interpretable CL methods include ICICLE, which mitigates explanation drift by preserving prototype–explanation consistency (Rymarczyk et al., 2023). IN2 constructs continual learning models directly upon human-understandable concepts, enabling concept-based knowledge representation and prediction (Yang et al., 2024b). CLG-CBM employs language-guided concept bottlenecks to learn human-understandable concepts across incremental tasks (Yu et al., 2025). CI-CBM combines concept regularization with pseudo-concept generation to preserve previously acquired concept knowledge without storing historical samples (Javadi et al., 2026). However, existing interpretable CL methods still face two major limitations, as illustrated in the left panel of Fig. 1. First, existing methods typically generate semantic concepts only once from class names within each incremental stage. Consequently, classification performance still largely depends on the training of data-driven classification networks, while the concepts serve primarily as auxiliary representations in the decision process. This results in limited adaptability of the semantic concept representations to the visual features present in class images. Second, final classification typically relies on trainable modules, such as neural classification heads and learnable mappings. Prior studies have shown that such trainable classification modules are subject to training-related issues such as classifier bias toward newly learned classes (Wu et al., 2019; Zhao et al., 2019), gradient imbalance (He & Zhu, 2024), task-recency bias (Rypeść et al., 2024), and sensitivity to task order (Lai et al., 2025a), thereby introducing optimization errors. As these modules are repeatedly optimized across incremental stages, the resulting optimization errors may progressively accumulate, degrading continual learning performance and potentially contributing to catastrophic forgetting. Overall, these limitations can be summarized as limited concept adaptability and optimization error accumulation, both closely related to the reliance on trainable classification modules for adapting representations and making classification decisions.

To address these limitations, we propose **Dynamic Concept Pool with Instance-Adaptive Gating (DyCoG)**. The core motivation behind DyCoG is to enable interpretable continual learning without relying on trainable classification modules for final classification, as illustrated in the right panel of Fig. 1. At a high level, by grounding both continual knowledge accumulation and classification in explicit semantic concepts, DyCoG provides two key advantages over existing interpretable CL methods: adaptive concept representations and robust statistical decisions. At the module level, DyCoG comprises two levels of concept modeling: class-level concept modeling and instance-level concept selection. The **Dynamic Concept Pool (DCP)** constructs a class-level concept space for inter-class discrimination, while the **Instance-Adaptive Concept Gate (IACG)** selects an instance-specific subset of concepts from the DCP for each input, providing instance-relevant discriminative evidence. Notably, IACG is trained separately for each optimized DCP rather than inherited across incremental stages, avoiding cross-stage accumulation of optimization errors. These advantages arise from two corresponding mechanisms. First, DCP performs feedback-guided concept refinement throughout the CL process, progressively updating semantic concepts using classification feedback from observed class images and thereby improving their adaptability to visual features relative to one-time concept generation. Second, DyCoG performs final classification directly using

statistics derived from concept representations, thereby avoiding optimization-error accumulation associated with repeated classifier training. Our contributions are summarized as follows:

- We propose the **Dynamic Concept Pool (DCP)**, which performs feedback-guided concept refinement by iteratively generating, evaluating, and updating explicit semantic concepts throughout continual learning. By maintaining knowledge in a human-understandable concept space, DCP enables adaptive and interpretable knowledge accumulation as the class space evolves.
- We introduce the **Instance-Adaptive Concept Gate (IACG)**, which selects an instance-specific subset of concepts from DCP for each input, transforming class-level concepts into instance-relevant semantic evidence. Together with direct statistical classification, IACG enables the final decision to be explicitly traced to the selected semantic concepts without relying on trainable classification modules for final classification.
- Extensive experiments on four benchmark datasets demonstrate that DyCoG achieves higher average incremental accuracy than the compared methods while remaining competitive in additional performance metrics and computational cost. More notably, as task partitions become finer, DyCoG exhibits an overall performance improvement rather than degradation, in contrast to the declining or saturating trends observed in most compared methods.

2 Related Work

Interpretable Continual Learning. ICICLE mitigates explanation drift through prototype-based interpretability regularization (Rymarczyk et al., 2023), while CIP-Net develops an exemplar-free, self-explainable prototype network (Di Valerio et al., 2026). Yang et al. (2024b) introduce IG-CL and IN2, which respectively employ interpretability-guided knowledge preservation and concept-based incremental learning. Among concept-based approaches, CONCIL updates concept and decision layers through recursive analytic learning (Lai et al., 2025b), while CLG-CBM employs language-guided concept bottlenecks to maintain semantic consistency across tasks (Yu et al., 2025). CI-CBM further integrates concept regularization and pseudo-concept generation to preserve interpretable knowledge without storing historical samples (Javadi et al., 2026).

Concept Refinement and Feedback-Driven Optimization. Existing concept-based methods improve concept quality through filtering and refinement. Kim et al. (2023) introduce visual activation scores to remove concepts with limited visual relevance, while VLG-CBM employs visual grounding to improve the faithfulness of concept annotations (Srivastava et al., 2024). In interpretable continual learning, CLG-CBM and CI-CBM incorporate concept selection or expansion as new tasks and classes arrive (Yu et al., 2025; Javadi et al., 2026). Beyond concept-based learning, feedback-driven iterative optimization offers a complementary approach to progressively improving candidate solutions. Self-Refine revises generated outputs through repeated feedback (Madaan et al., 2023), while ProTeGi uses textual critiques to iteratively optimize prompts (Pryzant et al., 2023). OPRO leverages previously evaluated solutions and their scores to guide subsequent generation (Yang et al., 2024a), whereas EvoPrompt and Promptbreeder evolve populations of candidate prompts through iterative generation, evaluation, and selection (Guo et al., 2024; Fernando et al., 2024).

3 Methodology

DyCoG combines class-level concept refinement through DCP with instance-level concept selection through IACG.

3.1 Dynamic Concept Pool (DCP)

A concept pool is a collection of semantic concepts used as explicit, interpretable features for classification.

As illustrated in Fig. 2, at each incremental stage t, DCP generates candidate concepts via an LLM API. In the r-th refinement round, the prompt contains four types of textual information: all observed classes, newly introduced classes, concepts retained from the previous round, and concepts

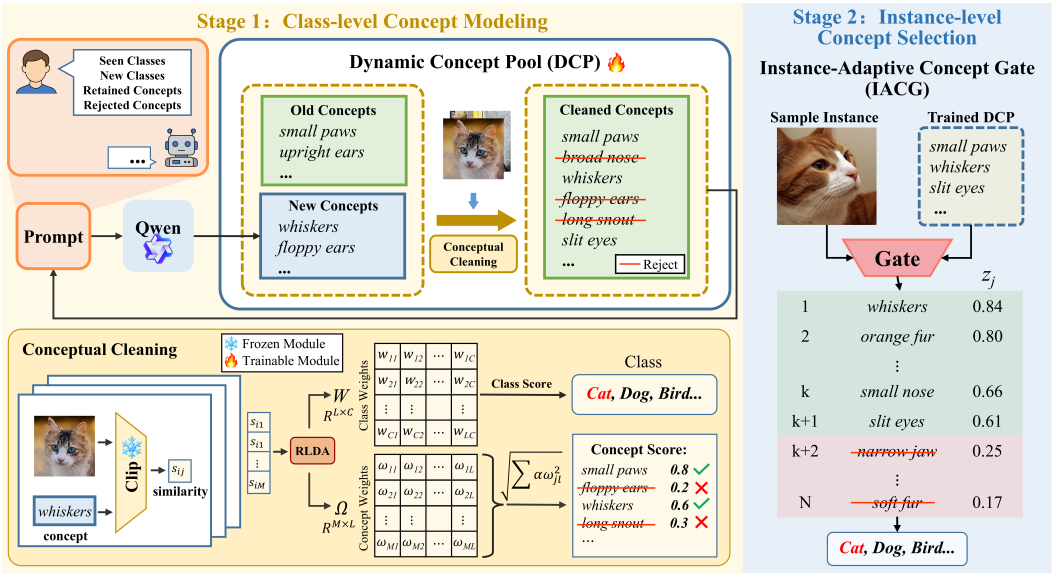

Figure 2: Overview of DyCoG. Stage 1 constructs class-level concepts in the Dynamic Concept Pool (DCP): an LLM proposes candidates, which are then retained or rejected by Conceptual Cleaning. Stage 2 applies the Instance-Adaptive Concept Gate (IACG) to select, for each input, a subset of DCP concepts that is used for final statistical classification.

rejected in earlier rounds of the current stage. Retained and rejected concepts guide complementary generation and reduce redundancy. The rejected-concept history is reset at the start of each stage. The complete prompting strategy is provided in Appendix A.1.

New candidates are merged with retained concepts and evaluated by Conceptual Cleaning using training images from all observed classes. Retained concepts enter the next round, while rejected concepts are recorded for subsequent prompting. Refinement stops when the improvement in classification accuracy remains below a predefined threshold for consecutive rounds (Appendix A.2).

For Conceptual Cleaning, we compute the similarity between a training image x_i and a concept c_j as $s_{ij} = \langle \bar{f}_v(x_i), \bar{f}_t(c_j) \rangle$ using frozen CLIP encoders (Radford et al., 2021), where $\bar{f}_v(\cdot)$ and $\bar{f}_t(\cdot)$ denote the ℓ_2-normalized visual and textual features from the frozen CLIP encoders. For N training images and M concepts, the response matrix $S \in \mathbb{R}^{N \times M}$ is used to fit a regularized linear discriminant analysis (RLDA) classifier (Friedman, 1989), whose parameters are derived from concept statistics without iterative optimization.

The classifier projects each M-dimensional concept-similarity representation into an L-dimensional class-discriminative space by minimizing within-class scatter while maximizing between-class scatter. Let $\Omega \in \mathbb{R}^{M \times L}$ denote the resulting concept weight matrix. The class scores for the i-th image are

$$g_i = s_i^\top \Omega W + b, \tag{1}$$

where $\mathbf{s}_i = [s_{i1}, \ldots, s_{iM}]^\top$, $W \in \mathbb{R}^{L \times C}$ denotes the class-score mapping matrix, and $\mathbf{b} \in \mathbb{R}^{1 \times C}$ denotes the class-dependent offset determined by the class statistics and prior probabilities. To quantify the contribution of each concept, we define the concept score of c_j as

$$q_j = \sqrt{\sum_{l=1}^{L} \alpha_l \omega_{jl}^2}, \tag{2}$$

where ω_{jl} denotes the coefficient of concept c_j in Ω, and α_l is the normalized non-negative weight derived from the corresponding generalized eigenvalue of the discriminant projection. The score q_j

measures the contribution of c_j to class discrimination (Perreault et al., 1979). Concepts with scores below τ are rejected, while the remaining concepts form the updated DCP.

DCP also supports human-guided concept addition, removal, and revision, as illustrated in Appendix A.3.

3.2 Instance-Adaptive Concept Gate (IACG)

Given the optimized DCP, IACG selects concepts for each input without directly predicting class labels.

The gating network takes CLIP-derived features from the input image and the optimized DCP as input and outputs a concept-importance vector $\mathbf{z} \in [0,1]^M$, where a larger z_j indicates a greater contribution of concept c_j to classifying the current input. Concepts with $z_j \geq 0.5$ are retained, while the remaining concepts are filtered out.

Since hard binarization of the concept-importance scores is non-differentiable, we employ the Straight-Through Estimator (STE) (Bengio et al., 2013) to enable gradient propagation through the discrete concept-selection operation during training. The binary concept-selection mask is defined as

$$Q = \mathrm{stopgrad}(\mathrm{step}(\mathbf{z} - 0.5) - \mathbf{z}) + \mathbf{z}, \tag{3}$$

where $\mathrm{stopgrad}(\cdot)$ blocks gradients through the hard selection operation while allowing gradients to propagate through $\mathbf{z}$. Applying the mask yields the instance-level concept representation $\hat{s}_i = s_i \odot Q$, where $\odot$ denotes element-wise multiplication. The same statistical classifier then computes $\hat{g}_i = \hat{s}_i^\top \Omega W + b$.

The gating objective consists of a classification loss and a binarization regularization term:

$$\mathcal{L}_{\mathrm{gate}} = \mathcal{L}_{\mathrm{cls}} + \lambda_{\mathrm{bin}} \mathcal{L}_{\mathrm{bin}}, \tag{4}$$

where λ_{bin} controls the regularization strength. The first term is the cross-entropy loss $\mathcal{L}_{\mathrm{cls}} = \mathrm{CrossEntropy}(\hat{g}_i, y_i)$, which supervises classification using the selected concepts. The second term encourages gate scores toward 0 or 1:

$$\mathcal{L}_{\mathrm{bin}} = \frac{1}{M} \sum_{j=1}^{M} z_j (1 - z_j). \tag{5}$$

The complete optimization and inference procedure is summarized in Algorithm 1 in Appendix A.4.

4 Experiments

4.1 Experimental Setup

Datasets and Protocols. We evaluate DyCoG on CIFAR100 (Krizhevsky & Hinton, 2009), Tiny-ImageNet (Le & Yang, 2015), FLOWER (Nilsback & Zisserman, 2008), and CUB (Wah et al., 2011) under three class-incremental learning settings with $T \in \{10, 20, 100\}$, where T denotes the number of tasks. Tasks contain disjoint sets of new classes, no task identity is provided at inference, and the same class order is used across different T for each dataset.

Baselines. We compare DyCoG with three groups of methods: General CIL (L2P, CODA-Prompt, AREA, FlyPrompt, and HippoTune (Wang et al., 2022; Smith et al., 2023a; Xie et al., 2026; Yan et al., 2026; Chen et al., 2026)), large-model-assisted CIL (GMM and ENGINE (Cao et al., 2024; Zhou et al., 2025)), and interpretable CIL (ICICLE, IN2, CLG-CBM, and CI-CBM (Rymarczyk et al., 2023; Yang et al., 2024b; Yu et al., 2025; Javadi et al., 2026)). All baselines follow their public implementations and recommended settings, using the same class order for comparison.

Table 1: Comparison of continual learning methods on four benchmark datasets under different task partitions. We report Average Incremental Accuracy ($\bar{A}$) for $T \in \{10, 20, 100\}$. Best and second-best results are shown in **bold** and underlined, respectively. "DCP" denotes classification using only the class-level concepts, while "DCP + IACG" denotes classification using instance-level concepts selected from DCP for each instance.

Category	Method	Source	CIFAR100			Tiny-ImageNet			FLOWER			CUB		
			$T=10$	$T=20$	$T=100$	$T=10$	$T=20$	$T=100$	$T=10$	$T=20$	$T=100$	$T=10$	$T=20$	$T=100$
General CIL	L2P	CVPR'22	86.98	82.69	0.52	75.27	71.48	46.15	89.36	81.90	6.80	59.22	60.60	26.00
	CODA-Prompt	CVPR'23	83.01	76.45	6.62	82.36	79.68	59.72	85.07	85.59	9.02	62.38	58.47	33.71
	AREA	ICML'26	81.75	86.31	85.39	78.11	74.17	80.16	97.37	96.95	96.43	87.99	88.19	87.27
	FlyPrompt	ICLR'26	88.86	86.21	4.93	85.49	84.02	72.30	93.74	86.78	9.91	72.67	67.24	63.76
	HippoTune	ICLR'26	90.05	88.19	4.08	87.65	85.94	73.27	96.59	96.57	10.14	82.38	81.56	73.70
LLM CIL	GMM	CVPR'24	78.56	77.38	71.06	74.97	73.08	73.90	79.80	75.50	55.99	37.68	35.67	31.84
	ENGINE	ICCV'25	86.36	86.55	85.52	81.62	82.01	81.69	92.99	89.57	88.96	87.39	86.61	79.23
Interpretable CIL	ICICLE	ICCV'23	34.86	21.15	5.43	23.38	16.51	5.35	12.86	10.99	5.58	32.23	23.94	3.86
	IN2	TMLR'24	24.48	16.10	5.15	21.53	14.84	4.29	20.21	14.90	5.04	8.46	7.29	4.87
	CLG-CBM	CVPR'25	84.02	83.74	83.15	81.11	81.14	79.55	95.19	91.97	95.61	85.35	86.11	85.29
	CI-CBM	TMLR'26	69.70	68.97	65.36	51.78	51.70	48.55	82.59	83.87	81.69	67.75	67.58	66.27
DyCoG	**DCP**	–	88.18	89.05	89.80	85.73	86.87	87.53	**99.10**	**99.32**	**99.44**	**90.56**	**90.89**	**91.98**
	DCP+IACG	–	**90.66**	**90.94**	**91.04**	**88.99**	**89.09**	**89.30**	**99.10**	**99.32**	**99.44**	**90.56**	**90.89**	**91.98**

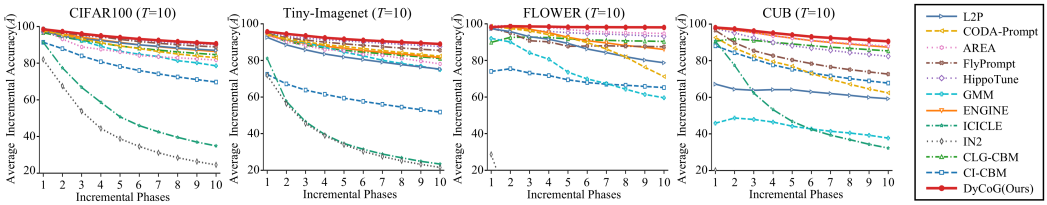

Figure 3: Average Incremental Accuracy ($\bar{A}$) trajectories under $T = 10$ on CIFAR100, Tiny-ImageNet, FLOWER, and CUB. DyCoG remains consistently stronger than the compared methods as incremental tasks arrive, especially in later stages.

Evaluation Metrics. We use average incremental accuracy $\bar{A} = \frac{1}{T}\sum_{t=1}^{T} A_t$ as the primary evaluation metric (Rebuffi et al., 2017; Chaudhry et al., 2018), where A_t denotes the accuracy after incremental stage t.

Implementation Details. We use Qwen3-Max-Preview (Yang et al., 2025) for concept generation, with up to 20 candidates generated per refinement round. RLDA uses shrinkage and ridge coefficients of 0.2 and 10^{-6}, respectively. IACG is optimized with AdamW (Loshchilov & Hutter, 2019) for 20 epochs with a learning rate of 2×10^{-4}.

4.2 Comparative Experiments

Table 1 and Fig. 3 show that DyCoG achieves the highest average incremental accuracy across all evaluated settings and maintains its advantage through later stages, consistently outperforming general CIL, large-model-assisted CIL, and existing interpretable CIL methods. DyCoG also remains competitive in additional performance metrics and computational cost, as detailed in Appendix A.5 and Appendix A.6.

Table 1 further compares DCP with DCP + IACG, which adds instance-level concept selection. IACG consistently improves average incremental accuracy on CIFAR100 and Tiny-ImageNet by identifying concept subsets more relevant to individual inputs, with further analysis provided in Appendix A.7. It provides no additional gain on FLOWER and CUB, a phenomenon analyzed in Sec. 4.4.

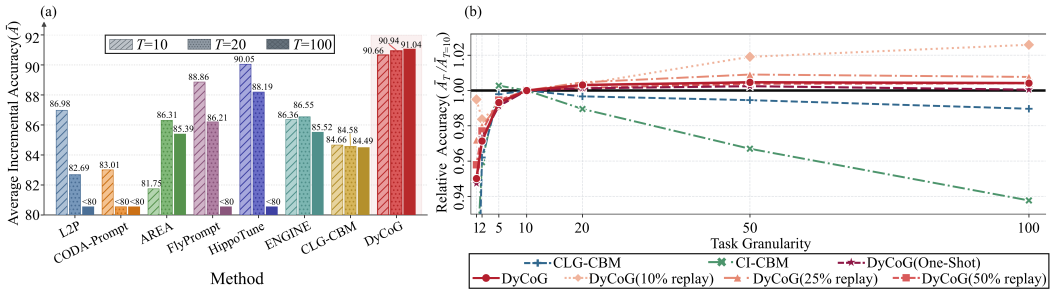

Figure 4: Performance trends on CIFAR100 under different task partitions. (a) Average Incremental Accuracy ($\bar{A}$) at $T \in \{10, 20, 100\}$. (b) Accuracy relative to the $T = 10$ result as task granularity increases. Ours denotes DyCoG.

Table 2: Average Incremental Accuracy ($\bar{A}$) on CIFAR100 from $T = 1$ to $T = 100$. The first three columns indicate one-shot concept generation, use of a trainable classification module, and the fraction of historical data retained. ↑ and ↓ mark an increase or decrease relative to the preceding task setting.

Method	1-Shot Concepts Generation	Trainable Classification Module	Historical Data Availability	T=1	T=2	T=5	T=10	T=20	T=50	T=100
CLG-CBM	✓	✓	–	70.61	80.83↑	83.84↑	84.02↑	83.74↓	83.56↓	83.15↓
CI-CBM	✓	✓	–	62.63	66.87↑	69.89↑	69.70↓	68.97↓	67.40↓	65.36↓
DyCoG	✓	×	100%	85.88	88.22↑	89.85↑	90.65↑	90.76↑	90.87↑	90.69↓
DyCoG	×	×	100%	86.14	88.07↑	90.04↑	90.66↑	90.94↑	91.08↑	91.04↓
DyCoG	×	×	50%	86.14	87.86↑	89.43↑	89.92↑	90.05↑	90.26↑	90.28↑
DyCoG	×	×	25%	86.14	86.84↑	87.91↑	88.62↑	89.03↑	89.42↑	89.30↓
DyCoG	×	×	10%	86.14	85.17↓	85.85↑	86.58↑	86.89↑	88.23↑	88.82↑

4.3 Performance Trends under Different Task Granularities

Table 3: Mean accuracy decrease per downward fluctuation for representative CIFAR100 classes.

Method	10th class: train				40th class: camel				70th class: trout			
	T=10	T=20	T=50	T=100	T=10	T=20	T=50	T=100	T=10	T=20	T=50	T=100
CLG-CBM	3.33	2.06	1.20	1.66	3.17	2.67	2.50	3.12	3.00	1.83	2.47	2.63
CI-CBM	5.89	3.61	2.33	2.43	3.00	3.67	2.80	3.12	2.67	2.67	2.33	2.37

Table 1 and Fig. 4 show that DyCoG generally improves its average incremental accuracy as the number of tasks T increases, despite the longer incremental sequences introduced by finer task partitions. To investigate this distinctive trend, we compare DyCoG with CLG-CBM and CI-CBM on CIFAR100 under $T \in \{1, 2, 5, 10, 20, 50, 100\}$, as reported in Table 2. Unlike DyCoG, both CLG-CBM and CI-CBM initially benefit from finer task partitions, but their performance subsequently declines as T increases further.

We attribute the trends of CLG-CBM and CI-CBM to two competing effects. First, finer partitions introduce fewer new classes per stage, reducing the difficulty of individual incremental updates. As T increases from 1 to 10, the mean accuracy when each class is first introduced rises from 70.61% to 90.16% for CLG-CBM and from 62.63% to 72.60% for CI-CBM, with complete per-class results in Appendix A.8. Second, finer partitions require more frequent re-optimization of trainable classification modules. Fig. 5 reveals more frequent fluctuations under finer partitions. Although accuracy may partially recover after individual updates, repeated downward fluctuations over a longer incremental sequence can offset the benefit of simpler per-stage learning. Table 3 further shows that the mean decrease per downward fluctuation remains broadly comparable across T, suggesting that the stronger degradation at large T is primarily associated with repeated error accumulation rather than larger individual drops.

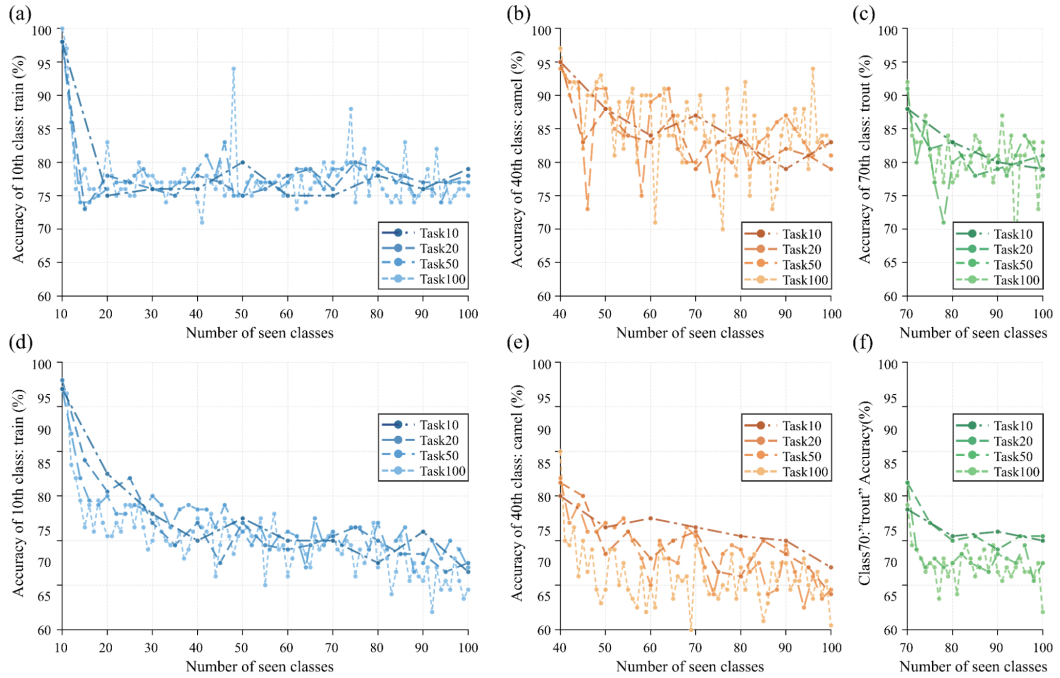

Figure 5: Accuracy of representative classes as the number of seen classes increases under different task partitions. The first row is CLG-CBM and the second row is CI-CBM.

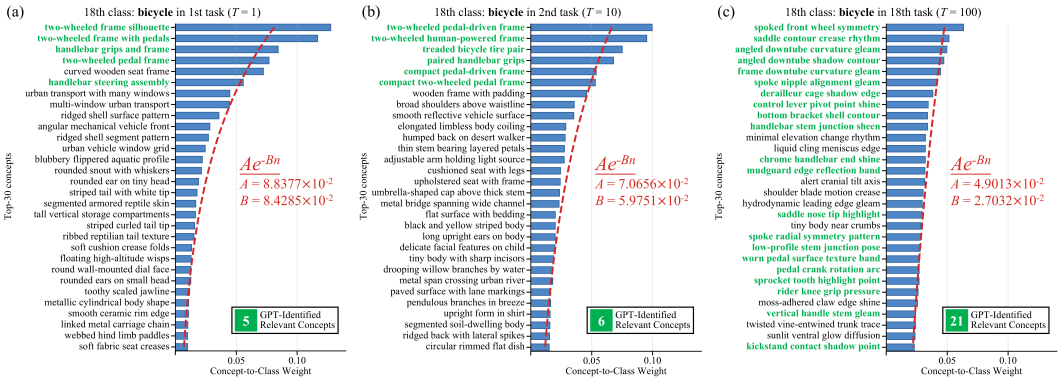

Figure 6: Top-ranked concepts for the bicycle class on CIFAR100 when the class is first introduced. (a) $T = 1$. (b) $T = 10$. (c) $T = 100$. Green labels indicate relevant concepts, and red dashed curves show exponential fits to the ranked concept-to-class weights.

DyCoG retains the benefit of simpler incremental stages while avoiding the cross-stage accumulation of optimization errors associated with repeatedly training a final classification module. As finer partitions provide more opportunities to refine the concept space, DCP can progressively capture richer discriminative semantic information. Fig. 6 illustrates this evolution using bicycle, the 18th class in the CIFAR100 sequence. Among the top-ranked concepts, the number identified as relevant by GPT increases from 5 to 6 and then to 21 as T increases. We further fit the ranked concept-to-class weights with Ae^{-Bn}. Both A and B decrease under finer partitions, indicating reduced dominance of the highest-weight concepts and slower weight decay across the ranking. These observations suggest that discriminative evidence is distributed over a larger set of fine-grained concepts, helping explain DyCoG's improvement with finer task partitions. The ablations in Table 2 further show that feedback-guided refinement and historical-data availability affect overall accuracy, while the broad improvement trend is largely preserved.

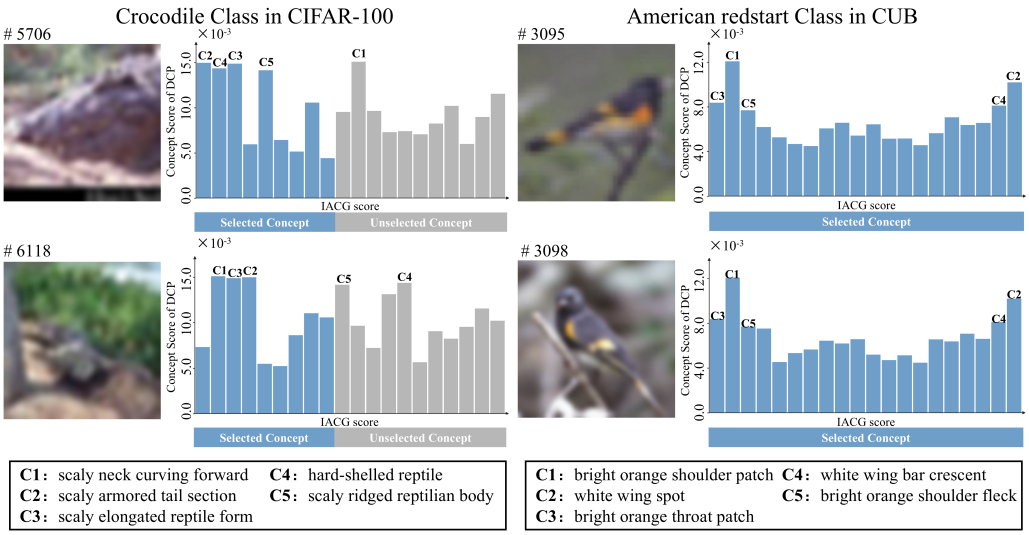

Figure 7: Concept-level interpretation of DyCoG on CIFAR100 and CUB. Bars show DCP concept scores ordered by IACG scores, with blue and gray denoting selected and unselected concepts.

4.4 Analysis of Interpretability

To analyze the interpretability of the two-level concept modeling in DyCoG, Fig. 7 shows the class-level concept importance learned by DCP and the instance-level concept selection performed by IACG. Each bar represents a concept, with its height indicating the DCP concept score. Concepts are ordered according to their IACG scores, and the selected concepts are highlighted in blue.

On CIFAR100, different instances from the same crocodile class exhibit different selected concept subsets, showing that IACG adapts the class-level concept representation to individual inputs. For example, concept C1 ("scaly neck curving forward") is retained as a valid class-level concept, but is filtered out for instance #5706 because the corresponding visual structure is not visible in the image. This illustrates how DCP provides class-level semantic evidence, while IACG further specializes it at the instance level.

In contrast, the two American redstart instances in CUB exhibit highly similar IACG rankings, and nearly all class-level concepts are retained. This indicates that IACG becomes functionally inactive when the DCP concept space is already compact and instances within the same class exhibit high visual consistency. PCA analysis of this compactness is provided in Appendix A.9.

5 Conclusion

In this work, we propose Dynamic Concept Pool with Instance-Adaptive Gating (DyCoG), which combines feedback-guided class-level concept refinement with instance-specific concept selection for direct statistical classification. DCP progressively updates explicit class-level concepts through feedback as new classes are introduced, while IACG selects an instance-specific subset that connects each prediction to relevant semantic evidence. Because the selected concept representation is classified statistically, the framework avoids repeatedly optimizing a final classification module across incremental stages. Across four datasets and multiple task partitions, DyCoG achieves the highest average incremental accuracy while remaining competitive in additional metrics and computational cost. The comparative and ablation results further show the complementary roles of class-level refinement and instance-level selection. Its overall improvement under finer task partitions further demonstrates the effectiveness and adaptability of dynamic concept modeling for interpretable continual learning.

AI USE STATEMENT

Generative AI tools were used solely to assist with preliminary literature research and English-language editing and polishing.

REFERENCES

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation, 2013. arXiv preprint arXiv:1308.3432.

Xusheng Cao, Haori Lu, Linlan Huang, Xialei Liu, and Ming-Ming Cheng. Generative Multi-modal Models are Good Class-Incremental Learners, 2024. arXiv preprint arXiv:2403.18383.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. In *Computer Vision - ECCV 2018*, pp. 556–572. Springer International Publishing, 2018. doi: 10.1007/978-3-030-01252-6_33.

Yanxi Chen, Xiuxing Li, Yuyang Han, Zhuo Wang, Qing Li, Ziyu Li, Xiang Li, Chen Wei, and Xia Wu. HippoTune: A Hippocampal Associative Loop-Inspired Fine-Tuning Method for Continual Learning. In *ICLR*, 2026.

MohammadReza Davari, Nader Asadi, Sudhir Mudur, Rahaf Aljundi, and Eugene Belilovsky. Probing Representation Forgetting in Supervised and Unsupervised Continual Learning, 2022. arXiv preprint arXiv:2203.13381.

Federico Di Valerio, Michela Proietti, Alessio Ragno, and Roberto Capobianco. CIP-Net: Continual Interpretable Prototype-based Network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(25):20772–20780, 2026.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. DyTox: Transformers for Continual Learning With DYnamic TOken eXpansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9285–9295, 2022.

Mohamed Elsayed and A. Rupam Mahmood. Addressing Loss of Plasticity and Catastrophic Forgetting in Continual Learning, 2024. arXiv preprint arXiv:2404.00781.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 13481–13544. PMLR, 2024.

Jerome H. Friedman. Regularized Discriminant Analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Evoprompt: Connecting LLMs with evolutionary algorithms yields powerful prompt optimizers. In *International Conference on Learning Representations*, 2024.

Jiangpeng He and Fengqing Zhu. Gradient Reweighting: Towards Imbalanced Class-Incremental Learning, 2024. arXiv preprint arXiv:2402.18528.

Amirhosein Javadi, Tuomas Oikarinen, Tara Javidi, and Tsui-Wei Weng. CI-CBM: Class-Incremental Concept Bottleneck Model for Interpretable Continual Learning, 2026. arXiv preprint arXiv:2604.14519.

Injae Kim, Jongha Kim, Joonmyung Choi, and Hyunwoo J. Kim. Concept bottleneck with visual concept filtering for explainable medical image classification, 2023.

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

Guannan Lai, Yujie Li, Xiangkun Wang, Junbo Zhang, Tianrui Li, and Xin Yang. Order-Robust Class Incremental Learning: Graph-Driven Dynamic Similarity Grouping, 2025a. arXiv preprint arXiv:2502.20032.

Songning Lai, Mingqian Liao, Zhangyi Hu, Jiayu Yang, Wenshuo Chen, Hongru Xiao, Jianheng Tang, Haicheng Liao, and Yutao Yue. Learning New Concepts, Remembering the Old: Continual Learning for Multimodal Concept Bottleneck Models, 2025b. arXiv preprint arXiv:2411.17471.

Giulia Lanzillotta, Sidak Pal Singh, Benjamin F. Grewe, and Thomas Hofmann. Towards guarantees for parameter isolation in continual learning, 2023. arXiv preprint arXiv:2310.01165.

Y. Le and X. Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Haoran Li, Jingfeng Wu, and Vladimir Braverman. Fixed Design Analysis of Regularization-Based Continual Learning, 2024. arXiv preprint arXiv:2303.10263.

Huiwei Lin, Baoquan Zhang, Shanshan Feng, Xutao Li, and Yunming Ye. PCR: Proxy-based Contrastive Replay for Online Class-Incremental Continual Learning, 2023. arXiv preprint arXiv:2304.04408.

Luca Salvatore Lorello, Marco Lippi, and Stefano Melacci. Continual Learning for Unsupervised Concept Bottleneck Discovery. In *Conference on Lifelong Learning Agents*, volume 274 of *Proceedings of Machine Learning Research*. PMLR, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, 2019. arXiv preprint arXiv:1711.05101.

Aojun Lu, Hangjie Yuan, Tao Feng, and Yanan Sun. Rethinking the Stability-Plasticity Trade-off in Continual Learning from an Architectural Perspective. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 40888–40902. PMLR, 2025.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023.

Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-Incremental Learning: Survey and Performance Evaluation on Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2023. doi: 10.1109/TPAMI.2022.3213473.

Michael McCloskey and Neal J. Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989. doi: 10.1016/S0079-7421(08)60536-8.

Maria-Elena Nilsback and Andrew Zisserman. Automated Flower Classification over a Large Number of Classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pp. 722–729, 2008. doi: 10.1109/ICVGIP.2008.47.

William D. Perreault, Douglas N. Behrman, and Gary M. Armstrong. Alternative approaches for interpretation of multiple discriminant analysis in marketing research. *Journal of Business Research*, 7(2):151–173, 1979. doi: 10.1016/0148-2963(79)90004-3.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, pp. 8748–8763, 2021.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental Classifier and Representation Learning, 2017. arXiv preprint arXiv:1611.07725.

Dawid Rymarczyk, Joost van de Weijer, Bartosz Zieliński, and Bartłomiej Twardowski. ICICLE: Interpretable Class Incremental Continual Learning. In *ICCV*, pp. 1887–1898, 2023.

Grzegorz Rypeść, Sebastian Cygert, Tomasz Trzciński, and Bartłomiej Twardowski. Task-recency bias strikes back: Adapting covariances in Exemplar-Free Class Incremental Learning, 2024. arXiv preprint arXiv:2409.18265.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. CODA-Prompt: COntinual Decomposed Attention-Based Prompting for Rehearsal-Free Continual Learning. In *CVPR*, pp. 11909–11919, 2023a.

James Seale Smith, Junjiao Tian, Shaunak Halbe, Yen-Chang Hsu, and Zsolt Kira. A Closer Look at Rehearsal-Free Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 2410–2420, 2023b.

Divyansh Srivastava, Ge Yan, and Tsui-Wei Weng. VLG-CBM: Training concept bottleneck models with vision-language guidance. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Ruilin Tong, Yuhang Liu, Javen Qinfeng Shi, and Dong Gong. Coreset Selection via Reducible Loss in Continual Learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011.

Zhenyi Wang, Yan Li, Li Shen, and Heng Huang. A Unified and General Framework for Continual Learning, 2024. arXiv preprint arXiv:2403.13249.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning To Prompt for Continual Learning. In *CVPR*, pp. 139–149, 2022.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large Scale Incremental Learning, 2019. arXiv preprint arXiv:1905.13260.

Zhen-Hao Xie, Yu-Cheng Shi, and Da-Wei Zhou. AREA: Attribute Extraction and Aggregation for CLIP-Based Class-Incremental Learning. In *ICML*, 2026.

Hongwei Yan, Guanglong Sun, Kanglei Zhou, Qian Li, Liyuan Wang, and Yi Zhong. FlyPrompt: Brain-Inspired Random-Expanded Routing with Temporal-Ensemble Experts for General Continual Learning. In *ICLR*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 Technical Report, 2025. arXiv preprint arXiv:2505.09388.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers, 2024a.

Sin-Han Yang, Tuomas Oikarinen, and Tsui-Wei Weng. Concept-Driven Continual Learning. *Transactions on Machine Learning Research*, 2024b.

Lu Yu, Haoyu Han, Zhe Tao, Hantao Yao, and Changsheng Xu. Language Guided Concept Bottleneck Models for Interpretable Continual Learning, 2025. arXiv preprint arXiv:2503.23283.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shutao Xia. Maintaining Discrimination and Fairness in Class Incremental Learning, 2019. arXiv preprint arXiv:1911.07053.

Xuyang Zhao, Huiyuan Wang, Weiran Huang, and Wei Lin. A Statistical Theory of Regularization-Based Continual Learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61021–61039. PMLR, 2024.

Da-Wei Zhou, Kai-Wen Li, Jingyi Ning, Han-Jia Ye, Lijun Zhang, and De-Chuan Zhan. External Knowledge Injection for CLIP-Based Class-Incremental Learning, 2025. arXiv preprint arXiv:2503.08510.

A Appendix

A.1 Prompt for Candidate Concept Generation

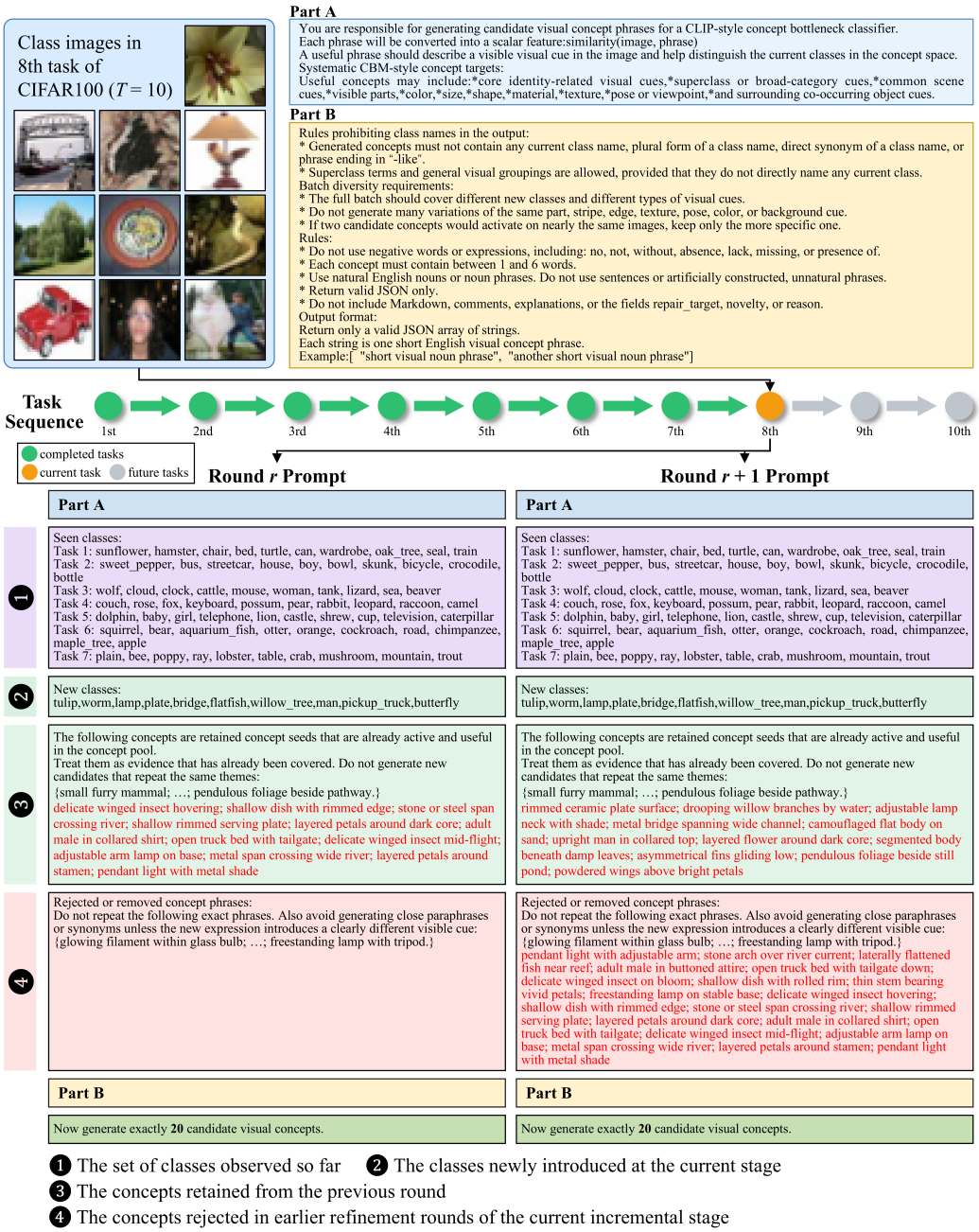

Figure A.1.1: Example prompt for the 8th task of CIFAR100 under $T = 10$, shown at two consecutive refinement rounds.

DCP uses a structured prompt with shared instructions and dynamically updated task-specific context (Fig. A.1.1). The shared instructions specify visually observable and diverse concepts, prohibit direct use of class names, and define the output format. Part A lists observed and newly introduced classes together with concepts retained and rejected in the current stage. These positive and negative contexts encourage complementary proposals and reduce redundancy. Part B requests $K = 20$

candidates per round. Refinement follows the stopping criterion in Appendix A.2, and the rejected-concept history is reset when the next stage begins.

A.2 Concept Evolution in DCP

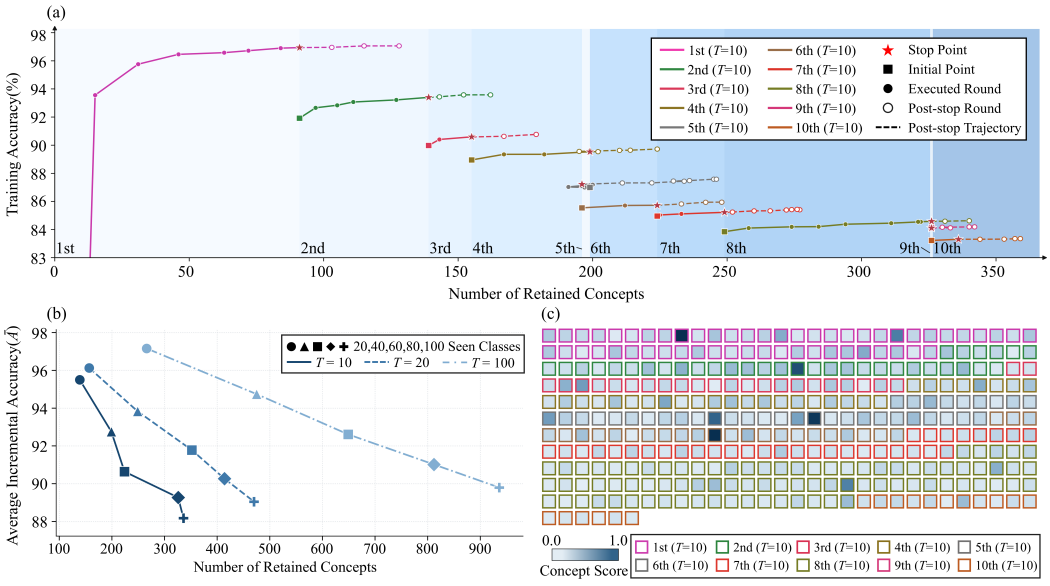

Figure A.2.1: Concept evolution of DCP on CIFAR100. (a) Retained concepts and statistical classification accuracy on the training set under $T = 10$. (b) Concept-pool size and Average Incremental Accuracy under different task partitions. (c) Final concept scores under $T = 10$.

Under $T = 10$, DCP progressively accumulates useful concepts across incremental stages. Fig. A.2.1(a) shows the retained-concept count and the statistical classifier's training accuracy. Each stage allows at most 15 refinement rounds and terminates early when accuracy improvement remains below 0.05% for two consecutive rounds. Additional rounds beyond this stopping point yield only marginal gains, supporting the effectiveness of the stopping criterion.

Finer task partitions generally lead to larger concept pools and higher average incremental accuracy (Fig. A.2.1(b)). DCP can discover more fine-grained useful concepts when fewer new classes are introduced per stage. This observation is consistent with the performance trend discussed in Sec. 4.3.

The final DCP under $T = 10$ contains 336 retained concepts (Fig. A.2.1(c)). Each square represents one concept ordered by generation time. The fill color indicates its final concept score, and the border color indicates the stage in which it was generated. Earlier concepts generally achieve higher final scores and, having survived more rounds of Conceptual Cleaning, tend to be more important for class discrimination.

A.3 Human-Guided Concept Intervention

The explicit semantic representation of DCP enables direct inspection and editing of class-level knowledge. Fig. A.3.1 illustrates concept deletion, revision, and addition: removing misleading concepts, correcting inaccurate descriptions, or introducing missing semantic cues can modify the concept space and improve classification performance. These interventions are used solely for interpretability analysis and are not involved in the automated results reported in the main experiments. The figure also presents a failure-attribution case in which a guinea pig is misclassified as teddy because the concept "stuffed plush bear seams" receives the highest concept score. This demonstrates that classification errors can be traced to explicit semantic evidence rather than opaque internal representations. Overall, DCP provides an interpretable and editable concept space in which misleading or incomplete semantic knowledge can be directly identified and adjusted.



Figure A.3.1: Human-guided concept deletion, revision, addition, and attribution of a misclassification to its highest-scoring concept.

A.4 Algorithm of DyCoG

Algorithm 1 summarizes DCP refinement, IACG training, and inference. IACG uses an AdamW weight decay of 10^{-4} and $\lambda_{\text{bin}} = 0.01$.

Algorithm 1 DyCoG: Dynamic Concept Pool with Instance-Adaptive Gating

Require: Incremental datasets $\{\mathcal{D}_t\}_{t=1}^{T}$, frozen CLIP encoders, LLM, thresholds τ, ϵ.
Ensure: Optimized DCPs $\{\mathcal{C}_t\}_{t=1}^{T}$, trained IACGs $\{G_t\}_{t=1}^{T}$, and associated RLDA parameters.
Initialize $\mathcal{C}_0 \leftarrow \emptyset$.
for $t = 1$ to T **do**
Stage 1: DCP Refinement
  Initialize retained concepts $\mathcal{C} \leftarrow \mathcal{C}_{t-1}$ and rejected concepts $\mathcal{R} \leftarrow \emptyset$.
  repeat
    Generate candidates $\mathcal{A}$ using the LLM with seen and new classes, and retained and rejected concepts.
    Merge $\mathcal{V} \leftarrow \mathcal{C} \cup \mathcal{A}$, compute CLIP similarities and RLDA concept scores $\{q_j\}$.
    Update $\mathcal{C} \leftarrow \{c_j \in \mathcal{V} \mid q_j \geq \tau\}$, $\mathcal{R} \leftarrow \mathcal{R} \cup (\mathcal{V} \setminus \mathcal{C})$.
  until the ϵ-based early-stopping criterion is met or the round limit is reached
  Set $\mathcal{C}_t \leftarrow \mathcal{C}$ and fit its corresponding RLDA classifier F_t.
Stage 2: IACG Training
  Initialize a new IACG G_t for $\mathcal{C}_t$.
  For observed-class samples x_i, compute concept similarities $\mathbf{s}_i$ and predict importance scores $\mathbf{z}_i$ using G_t.
  Obtain binary mask $\mathbf{Q}_i$ via STE, $\hat{\mathbf{s}}_i = \mathbf{s}_i \odot \mathbf{Q}_i$.
  Train G_t on observed-class data using $\mathcal{L}_{\text{gate}}$, keeping F_t fixed.
end for
Inference (given input x at stage t)
Compute concept similarities $\mathbf{s} \leftarrow \text{Sim}(x, \mathcal{C}_t)$ and obtain $\hat{\mathbf{s}}$ using G_t.
return the class with the highest RLDA score using $\hat{\mathbf{s}}$.

A.5 Additional Performance Metrics

Table A.5.1: Final Accuracy (A_{last}) and Relative Forgetting (RF) on four datasets under $T \in \{10, 20, 100\}$. $\uparrow$ and $\downarrow$ indicate that higher and lower values are better, respectively. Best and second-best results are shown in **bold** and underlined, respectively.

Category	Method	Source	CIFAR100						Tiny-ImageNet						FLOWER						CUB					
			T=10		T=20		T=100		T=10		T=20		T=100		T=10		T=20		T=100		T=10		T=20		T=100	
			$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$	$A_{\text{last}}\uparrow$	$RF\downarrow$
General CIL	L2P	CVPR'22	81.15	8.17	74.02	16.44	1.00	99.00	64.84	9.11	59.96	15.29	34.73	43.22	82.76	8.17	70.58	17.92	1.39	98.57	52.50	22.70	50.72	18.82	24.91	53.78
	CODA-Prompt	CVPR'23	74.52	14.09	68.66	17.98	1.36	94.91	73.83	10.85	69.46	14.82	42.95	34.15	70.13	35.18	62.55	24.95	1.96	95.14	42.75	41.94	40.90	33.34	22.69	53.15
	AREA	ICML'26	76.55	6.04	79.89	8.86	77.71	9.73	64.42	16.05	57.96	17.39	70.97	9.98	96.44	1.12	95.82	7.01	95.22	1.08	82.17	**4.95**	81.31	6.60	79.53	8.98
	FlyPrompt	ICLR'26	83.80	5.57	79.77	7.41	1.17	74.38	79.76	6.74	76.92	7.66	63.14	15.35	92.93	2.87	85.28	7.61	2.15	94.60	61.96	16.80	53.50	25.80	50.66	39.04
	HippoTune	ICLR'26	**85.97**	4.94	82.26	8.73	0.58	80.71	82.38	7.53	80.02	9.17	60.71	25.87	94.88	2.10	91.46	2.28	2.39	91.59	73.05	11.11	71.31	14.46	62.13	20.48
LLM CIL	GMM	CVPR'24	63.76	37.39	58.50	41.78	63.76	35.69	58.23	37.34	55.67	40.04	60.22	35.90	71.56	26.51	65.78	33.00	58.56	40.83	23.54	67.75	20.64	73.47	18.48	76.14
	ENGINE	ICCV'25	79.32	11.42	79.32	9.89	77.59	7.27	73.37	10.30	73.19	11.76	72.74	10.46	87.54	7.59	78.06	14.13	79.59	11.53	80.17	6.70	77.65	8.27	62.96	21.18
Interpretable CIL	ICICLE	ICCV'23	15.95	74.29	7.19	87.37	1.05	98.67	10.09	77.04	3.52	81.53	0.60	98.23	4.94	88.37	2.93	95.00	1.24	98.17	13.35	76.30	10.53	80.02	0.51	98.85
	IN2	TMLR'24	7.85	89.59	5.34	93.97	0.64	98.71	6.38	89.99	3.42	94.91	0.91	99.98	7.55	89.09	5.36	93.80	1.00	99.00	3.36	93.00	2.35	94.75	0.95	98.55
	CLG-CBM	CVPR'25	78.24	12.98	78.10	12.48	76.80	17.88	73.42	15.61	73.06	17.81	71.02	22.35	94.36	2.40	94.11	2.34	94.50	4.06	79.22	6.35	79.50	8.42	77.46	15.21
	CI-CBM	TMLR'26	52.76	19.60	55.98	23.71	49.78	34.97	39.82	20.82	39.05	23.48	35.13	36.36	79.80	7.44	78.19	10.32	73.49	15.69	56.97	17.60	56.83	17.65	51.93	26.79
DyCoG	DCP	–	83.37	**4.84**	83.65	**5.59**	84.01	**6.52**	81.20	5.34	81.65	6.07	81.42	**7.39**	**98.91**	**0.33**	**99.12**	**0.31**	**99.11**	**0.35**	**85.05**	5.12	**85.52**	**4.95**	**86.00**	**5.80**
	DCP+IACG	–	85.53	5.49	**86.07**	5.80	**85.50**	8.45	**84.32**	**5.15**	**84.11**	**5.95**	**83.66**	8.46	**98.91**	**0.33**	**99.12**	**0.31**	**99.11**	**0.35**	**85.05**	5.12	**85.52**	**4.95**	**86.00**	**5.80**

In addition to average incremental accuracy, we further evaluate continual learning performance using Final Accuracy (A_{last}) and Relative Forgetting (RF). Final Accuracy denotes the classification accuracy over all observed classes after the final incremental stage.

To evaluate forgetting more fairly across methods with different accuracy levels, we introduce Relative Forgetting (RF). Conventional absolute forgetting measures the decrease from a task's historical accuracy to its final accuracy, which may make methods with relatively low historical accuracy appear to suffer less forgetting simply because their absolute performance drop is small. We therefore normalize the accuracy decrease by the corresponding historical accuracy. For task i, the relative forgetting is defined as

$$f_i^{\text{rel}} = \max_{t \in \{i, \ldots, T-1\}} \frac{a_{t,i} - a_{T,i}}{a_{t,i}}, \tag{6}$$

where $a_{t,i}$ denotes the test accuracy on task i after learning stage t, and $a_{T,i}$ denotes its accuracy after the final stage. RF is computed by averaging over the first $T-1$ tasks:

$$RF = \frac{1}{T-1} \sum_{i=1}^{T-1} f_i^{\text{rel}}. \tag{7}$$

The final task is excluded because no subsequent learning stage is available to induce forgetting. A lower RF indicates better relative preservation of previously learned knowledge.

Table A.5.1 reports A_{last} and RF on four datasets under $T \in \{10, 20, 100\}$. DyCoG maintains strong final-stage accuracy across datasets and task partitions, consistent with the average incremental accuracy results in Table 1, indicating that its advantage persists after the complete incremental sequence. It also exhibits consistently low RF, indicating effective preservation of previously acquired knowledge relative to historical performance. In particular, DCP maintains stable RF under increasingly fine task partitions on several datasets, whereas many compared methods exhibit substantially increased forgetting as T grows.

A.6 Computational Cost Analysis

We evaluate the computational cost of DyCoG in terms of runtime and LLM API usage. Table A.6.1 reports the average per-task runtime across datasets and task partitions. DyCoG maintains relatively low runtime, with modest overhead over DCP from IACG training. As shown in Fig. A.6.1, DCP and DyCoG exhibit substantially flatter runtime trends than the baselines as the average number of training images per task increases, indicating relatively low sensitivity to data scale, particularly on larger incremental datasets.

Table A.6.1: Average per-task runtime on four datasets under $T \in \{10, 20, 100\}$. Best and second-best results are shown in **bold** and underlined, respectively.

Category	Method	Source	CIFAR100			Tiny-ImageNet			FLOWER			CUB		
			T=10	T=20	T=100	T=10	T=20	T=100	T=10	T=20	T=100	T=10	T=20	T=100
General CIL	L2P	CVPR'22	513.69	549.91	1632.87	902.58	636.82	161.24	116.08	140.95	115.59	132.45	118.17	114.69
	CODA-Prompt	CVPR'23	922.25	504.53	155.44	1742.57	929.08	265.08	81.01	95.39	99.10	176.15	112.12	109.97
	AREA	ICML'26	2261.67	1844.82	226.30	4047.94	2011.67	437.28	115.85	83.40	70.59	269.24	156.67	87.51
	FlyPrompt	ICLR'26	248.63	136.92	50.52	480.69	245.73	71.31	29.08	23.77	19.67	41.96	29.24	20.23
	HippoTune	ICLR'26	342.69	187.77	83.08	636.40	338.99	126.85	37.88	36.90	59.85	75.05	57.14	56.92
LLM CIL	GMM	CVPR'24	609.40	939.80	391.17	1682.13	922.84	301.93	320.41	219.46	139.44	355.50	229.61	131.09
	ENGINE	ICCV'25	933.10	716.70	923.08	1568.30	1062.06	663.50	384.80	363.65	346.46	424.10	379.35	346.91
Interpretable CIL	ICICLE	ICCV'23	463.25	196.76	126.25	2069.25	481.26	250.71	93.83	76.66	43.50	1223.94	661.00	215.80
	IN2	TMLR'24	106.20	106.82	263.53	253.55	163.94	175.75	37.90	31.17	26.80	38.60	31.60	24.70
	CLG-CBM	CVPR'25	518.70	624.15	1370.44	854.10	1192.40	224.51	41.80	32.91	25.38	97.60	101.20	127.70
	CI-CBM	TMLR'26	126.80	120.40	124.91	199.30	197.50	178.34	**9.00**	**5.15**	**4.85**	**24.00**	**18.75**	**18.19**
DyCoG	DCP	–	**59.74**	**45.01**	**38.43**	**100.96**	**74.66**	**65.33**	39.87	37.48	40.23	46.30	42.87	41.11
	DyCoG	–	64.23	48.93	43.48	110.00	83.87	77.93	39.87	37.48	40.23	46.30	42.87	41.11

Table A.6.2: LLM API usage during training and inference. DyCoG results are shown in **bold**.

Method	Source	CIFAR100	Tiny-ImageNet	FLOWER	CUB
Training stage					
GMM	CVPR'24	58.14	64.82	65.79	67.91
ENGINE	ICCV'25	0	0	0	0
DyCoG	–	**3.33**	**1.65**	**44.54**	**22.73**
Inference stage					
GMM	CVPR'24	90.93	98.02	98.72	101.05
ENGINE	ICCV'25	146.51	497.8	242.36	1111.08
DyCoG	–	**0**	**0**	**0**	**0**

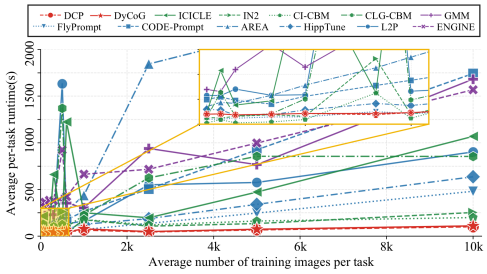

Figure A.6.1: Per-task runtime versus training images per task.

The additional runtime of DyCoG over DCP mainly comes from IACG training. In practical deployment, however, IACG only needs to be trained for the optimized DCP on which instance-level concept selection is required. In scenarios where instance-level selection is performed only after the final incremental stage, IACG can therefore be trained only once after continual knowledge accumulation is completed, making the practical continual-update cost closer to that of DCP.

Table A.6.2 reports LLM API usage during training and inference. During training, candidate concepts are generated from textual prompts without repeatedly uploading training images to the LLM, while feedback-guided stopping avoids unnecessary calls when further refinement yields little gain. Consequently, DyCoG incurs limited API cost and requires substantially fewer training-stage API calls than GMM on most evaluated datasets. At inference, DyCoG requires no LLM API calls, as prediction relies on learned concept representations and statistical classification, whereas GMM and ENGINE continue to require LLM calls. This makes DyCoG suitable for offline or resource-constrained deployment where external LLM access may be unavailable or undesirable.

A.7 Instance-Level Concept Selection by IACG

Fig. A.7.1(a) shows accuracy versus the mean number of selected concepts per image on CIFAR100. Accuracy first rises as more useful concepts are retained, then saturates or drops when less relevant concepts are included. IACG keeps concepts with gate scores above 0.5, reaching strong accuracy with far fewer concepts than the full class-level set.

Fig. A.7.1(b) compares class-level and instance-level representations. After instance-level selection, points shift toward the upper-left, indicating higher average incremental accuracy with fewer concepts. IACG thus filters less relevant concepts and yields a more compact representation for each input.

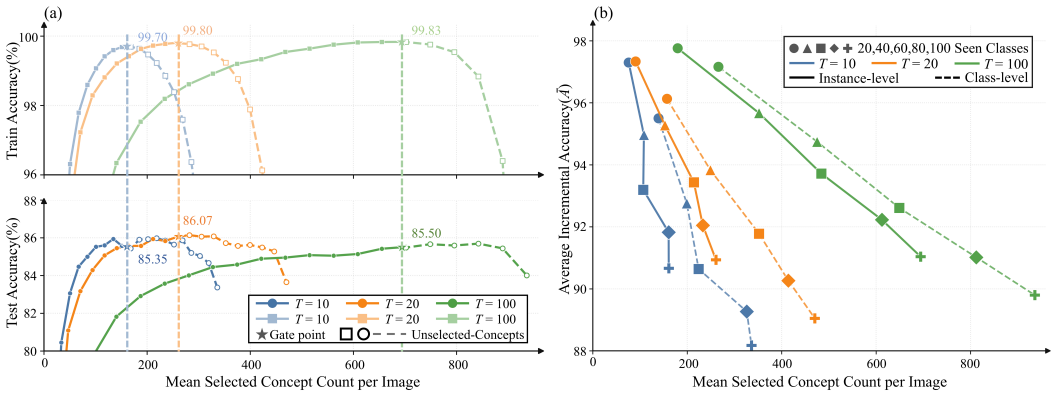

Figure A.7.1: Effect of instance-level concept selection by IACG on CIFAR100. (a) Accuracy versus the number of selected concepts. (b) Class-level and instance-level representations.

A.8 Initial Accuracy under Different Task Partitions

Fig. A.8.1 examines the first competing effect using CLG-CBM and CI-CBM on CIFAR100. For each class, we record its classification accuracy when it is first introduced. Although individual classes vary, the mean initial accuracy consistently increases as fewer new classes are introduced at each stage. Specifically, as T increases from 1 to 10, the mean rises from 70.61% to 90.16% for CLG-CBM and from 62.63% to 72.60% for CI-CBM. Because the measurement is taken when each class first appears, it separates the difficulty of the current stage from the cumulative effects of later updates. These results support the first effect discussed in Sec. 4.3, showing that finer task partitions simplify the per-stage classification problem and can initially improve continual learning performance. The consistent direction across both baselines shows that this effect is not specific to a single method. However, the improvement concerns only accuracy at class introduction and does not determine performance after the complete sequence. As analyzed in Sec. 4.3, more frequent re-optimization can offset this initial benefit as T grows. Fig. A.8.1 therefore isolates the short-term benefit of simpler stages, while Fig. 5 and Table 3 characterize the accumulated effect over the incremental sequence.

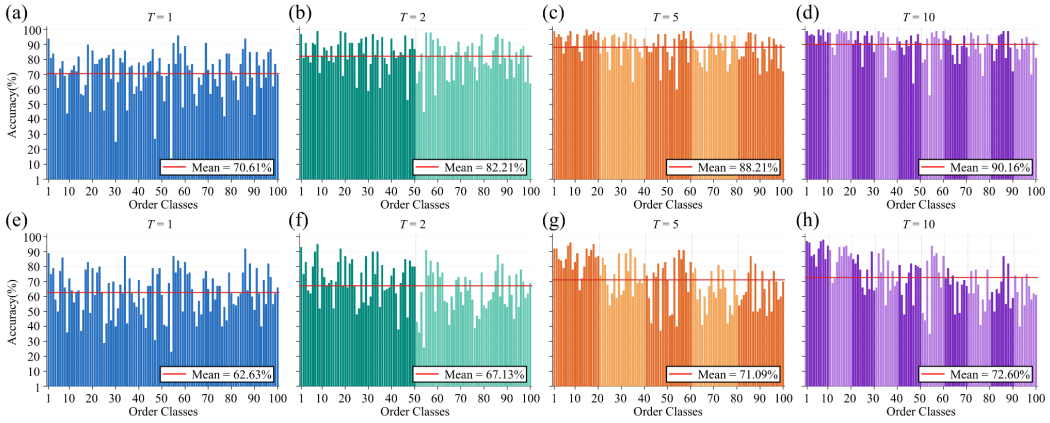

Figure A.8.1: Initial accuracy of each class when first introduced on CIFAR100 under different task partitions. The red line denotes the mean. The first row is CLG-CBM and the second is CI-CBM.

A.9 PCA Analysis of DCP Concept-Space Compactness

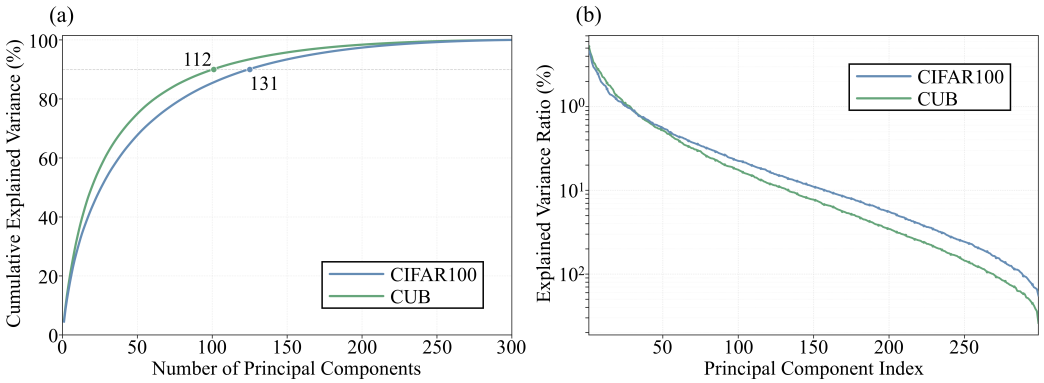

Figure A.9.1: PCA of the final DCP concept spaces on CIFAR100 and CUB. (a) Cumulative explained variance. (b) Explained variance ratio of each principal component.

Fig. A.9.1 compares the final DCP concept spaces on CIFAR100 and CUB. CUB reaches 90% cumulative explained variance with fewer principal components than CIFAR100 (112 vs. 131), and its explained variance ratio decays faster, indicating a more compact concept space. This compactness helps explain the functional inactivity of IACG on CUB, where lower semantic variation leaves little room for instance-level selection.