Class-Aware Drift Compensation for Non-Uniform Semantic Shift in Continual Learning (Appendix)

Fankang Xu[1], Lu Jin[1], Yanpeng Sun[2], Shiyu Xuan[1], Zechao Li[1] *
[1] School of Computer Science and Engineering, Nanjing University of Science and Technology
[2] Singapore University of Technology and Design
{frank.xu, lujin, yanpeng_sun, shiyu_xuan, zechao.li}@njust.edu.cn

1. Experimental Details

Implementation details. We employ ResNet-18, a commonly used architecture, as our backbone model, and train it from scratch following the CIL protocol. For CIFAR-100, we apply the same data augmentation strategies as those used for CIFAR-10 in the PyCIL framework [9], ensuring consistent treatment across all compared methods. To maintain standard training practices, random cropping and horizontal flipping are applied to the training data for all datasets, in line with prior work. The network is optimized using SGD with a momentum value of 0.9. During the initial task, training lasts for 200 epochs, starting with a learning rate of 0.1, which is reduced by a factor of 0.1 at epochs 60, 120, and 160. Each subsequent task undergoes 100 epochs of training, using learning rates of 0.05 for CIFAR-100, 0.005 for TinyImageNet, and 0.01 for ImageNet-Subset, with the same decay schedule applied every 45 epochs. The weight decay is configured as 5×10^{-4} for the initial task and 2×10^{-4} for later tasks. Batch sizes are fixed at 128 for CIFAR-100, and 64 for TinyImageNet and ImageNet-Subset.

The proposed method is outlined in Algorithm 1. The first task, which only involves learning from new data, is treated as an independent training phase (Lines 1–6). For subsequent tasks (Lines 8–21), training incorporates three loss terms. The cross-entropy loss $\mathcal{L}_{ce}$ in Line 15 serves as the baseline for learning new classes. The $\mathcal{L}_{trsf}$ loss in Line 18 models the transfer patterns between old and new feature spaces, while the drift-guided knowledge distillation (DGKD) in Line 17 leverages class-wise drift degrees τ, computed in Lines 11–13. After each task, class prototypes of current task are computed (Lines 7 and 25). For tasks beyond the first, class-aware drift compensation is applied to update the old class prototypes (Lines 22–24).

Datasets details. The CIFAR-100 dataset [3] comprises a total of 60,000 color images sized at 32 × 32 pixels, evenly distributed across 100 object categories. For each class, there are 500 samples for training and 100 for testing. Tiny-

Algorithm 1: Pseudocode of Training Process

Input: A sequence of tasks $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$, training data $\mathcal{D}_t^{train} = \{(x_i, y_i)\}_{i=1}^{n}$ of task T_t, prototypes $\mathrm{P}_{1:\mathrm{t}-1} = \{\mathrm{p}_i\}_{i=1}^{|\mathcal{C}_{1:t-1}|}$ for past classes, initial model parameters $\{\theta, \pi\}$ for feature extractor f_θ^0 and classifier g_π^0, initial parameters $\{\mathbf{W}, \mathbf{b}\}$ for drift estimator Ψ. CEL denotes the cross entropy loss, and MSE indicates the mean squared error. All equations are numbered according to the main text.
Output: Optimal model θ, π.

1: **while** epoch $<$ max_epoch **do**
2: **for** batch **in** training_batches **do**
3: $\mathcal{L}_{ce} \leftarrow \mathrm{CEL}\,(x, y)$ for new classes learning
4: update $\{\theta, \pi\}$ with loss $\mathcal{L}_{ce}$
5: **end for**
6: **end while**
7: build prototypes for each class of the current task T_1
8: **for** $T_t \in \{T_t\}_{t=2}^{N}$ **do**
9: $f_\theta^t \leftarrow f_\theta^{t-1}$, $g_\pi^t \leftarrow g_\pi^{t-1}$; freeze $\left\{f_\theta^{t-1}, g_\pi^{t-1}\right\}$
10: **while** epoch $<$ max_epoch **do**
11: model shift $\mathcal{S}^t\,(c)$ in Eq. (3) for each class
12: compute drift degree τ in Eq. (6) for each class
13: update τ as $\tau \leftarrow (\tau + \tau')\,/2$
14: **for** batch **in** training_batches **do**
15: $\mathcal{L}_{ce} \leftarrow \mathrm{CEL}\,(x, y)$
16: $\mathcal{L}_{trsf} \leftarrow \mathrm{MSE}(f_\theta^{t-1}(x)\mathbf{W} + \mathbf{b}, f_\theta^t(x))$ in Eq. (4)
17: $\mathcal{L}_{dgkd} \leftarrow \mathrm{CEL}\left(\tau, g_\pi^{t-1}\left(z\right), g_\pi^t\left(z\right)\right)$ in Eq. (8)
18: $\mathcal{L} = \mathcal{L}_{\mathrm{ce}} + \gamma \cdot \mathcal{L}_{\mathrm{dgkd}} + \delta \cdot \mathcal{L}_{\mathrm{trsf}}$
19: update $\{\theta, \pi, \mathbf{W}, \mathbf{b}\}$ with overall loss $\mathcal{L}$
20: **end for**
21: **end while**
22: **for** prototype **in** $\mathrm{P}_{1:\mathrm{t}-1}$ **do**
23: $\mathrm{p} \leftarrow \Psi\left(\mathrm{p}_i\right) = \mathrm{p} + \tau \cdot (\mathrm{p} \cdot \mathbf{W} + \mathbf{b} - \mathrm{p})$ in Eq. (7)
24: **end for**
25: build prototypes for each class of the current task T_t
26: **end for**

*Corresponding author.

Table A1. Comparison of EFCIL methods on CIFAR-100, TinyImageNet, and ImageNet-Subset under 20-task split. Results marked with † are reported from [7].

	Methods	CIFAR-100		TinyImageNet		ImageNet-Subset	
		$\mathcal{A}_{last}$	$\mathcal{A}_{inc}$	$\mathcal{A}_{last}$	$\mathcal{A}_{inc}$	$\mathcal{A}_{last}$	$\mathcal{A}_{inc}$
20 stages	LwF†	17.44	38.39	15.02	32.94	18.64	40.23
	PASS	25.31 ± 0.18	42.55 ± 0.82	18.73 ± 1.43	32.01 ± 1.68	22.45 ± 1.11	39.55 ± 0.60
	SSRE	9.97 ± 0.35	22.23 ± 0.15	10.69 ± 1.20	21.80 ± 2.37	16.25 ± 1.05	31.15 ± 1.53
	FeTrIL	25.17 ± 2.57	42.69 ± 2.49	11.76 ± 0.22	22.66 ± 1.31	20.74 ± 1.24	36.63 ± 1.43
	FeCAM†	23.82	38.51	25.36	38.59	26.03	40.87
	NAPA-VQ	23.43 ± 0.39	37.77 ± 0.01	13.53 ± 0.02	22.77 ± 0.11	12.28 ± 0.14	24.38 ± 0.03
	FKSR	18.66 ± 0.70	35.26 ± 0.52	14.93 ± 2.63	25.65 ± 3.35	26.60 ± 0.98	41.27 ± 0.71
	FCS	13.16 ± 7.72	29.84 ± 7.61	13.77 ± 0.76	23.41 ± 0.55	26.61 ± 0.49	41.28 ± 0.34
	EFC	36.97 ± 0.95	51.66 ± 1.60	28.69 ± 0.40	42.07 ± 0.96	35.75 ± 1.74	49.92 ± 2.05
	Ours	**37.72 ± 0.62**	**54.94 ± 0.78**	**30.03 ± 0.50**	**44.24 ± 1.25**	**42.41 ± 0.91**	**59.58 ± 0.39**

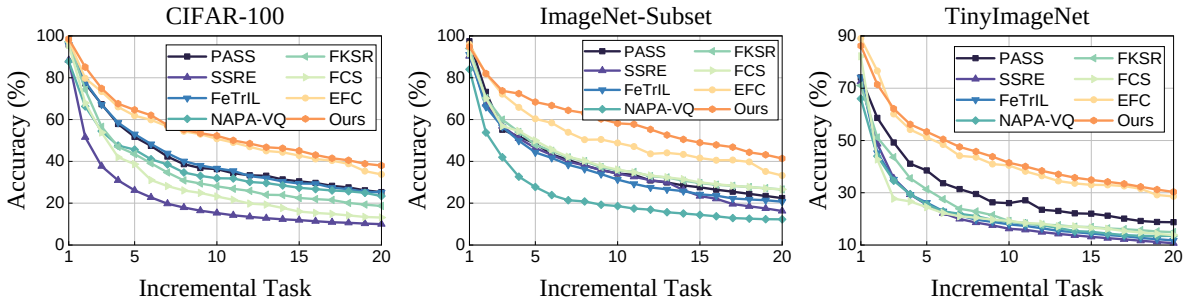

Figure A1. Accuracy trends after each task for different methods on CIFAR-100, TinyImageNet, and ImageNet-Subset under 20-task split.

ImageNet [**?**], on the other hand, contains 200 classes, with each class including 500 training examples, 50 for validation, and 50 for testing. All images in this dataset are resized to 64 × 64 pixels. Its finer granularity and class expansion across different stages make it particularly suitable for testing the adaptability of incremental learning algorithms. The ImageNet-Subset is constructed by randomly sampling 100 categories from the full ImageNet-1k dataset [1], using random seed 1993. Each category in this subset includes roughly 1,300 training images and 50 for testing, with images scaled to 256 × 256 pixels, offering a significantly higher resolution than CIFAR-100 and Tiny-ImageNet.

Evaluation metrics. We adopt two commonly used metrics in CIL, including the last task accuracy $\mathcal{A}_{last}$ and average incremental accuracy $\mathcal{A}_{inc}$. Let acc_i denote the classification accuracy over all seen classes after completing task T_i. Then, the last accuracy is defined as $\mathcal{A}_{last} = acc_N$, and the average incremental accuracy is computed as $\frac{1}{N}\sum_{i=1}^{N} acc_i$, where N denotes the index of the last task.

2. Additional Experimental Results

Results under the 20-task split. To evaluate the effectiveness of the proposed method in exemplar-free class incremental learning (EFCIL), we conduct comprehensive comparisons on three benchmark datasets, including CIFAR-100, TinyImageNet, and ImageNet-Subset, under the challenging 20-task split. As shown in Table A1, our approach consistently outperforms existing state-of-the-art methods across all evaluation metrics, achieving the highest accuracy in both last task accuracy ($\mathcal{A}_{last}$) and average incremental accuracy ($\mathcal{A}_{inc}$). On CIFAR100, our method achieves an $\mathcal{A}_{last}$ of 37.72% and an $\mathcal{A}_{inc}$ of 54.94%, surpassing the strongest baseline EFC by 0.75% and 3.28%, respectively. Similar improvements can be observed on TinyImageNet, where our approach yields 30.03% in $\mathcal{A}_{last}$ and 44.24% in $\mathcal{A}_{inc}$, outperforming the second best method, EFC, by 1.34% and 2.17%, respectively. The performance advantage is further amplified on ImageNetSubset, where our method exceeds the previous best results from EFC by 6.66% in $\mathcal{A}_{last}$ and 9.66% in $\mathcal{A}_{inc}$, respectively. These consistent gains validate the robustness and scalability of our method in addressing catastrophic forgetting under exemplar-free settings. Although EFC [6] achieves competitive results by leveraging a prototype drift compensation mechanism based on the empirical feature matrix within the SDC framework [8], it treats all classes equally during the compensation process. This uniform treatment inevitably leads to accumulated errors over time, causing the prototypes to deviate from their oracle ones as incre-

mental tasks proceed. In contrast, our method introduces class-aware compensation strategies that explicitly account for the varying degrees of semantic shift experienced by different old classes during task transitions. By modeling the class-wise semantic shift more precisely, our approach achieves superior performance across all benchmarks.

Fig. A1 illustrates the test accuracy trends after each incremental task on CIFAR100, TinyImageNet, and ImageNetSubset. Across all three datasets, our method consistently maintains a higher level of accuracy throughout the entire learning process compared to other baselines. Notably, while other methods exhibit a sharp accuracy drop in early stages followed by gradual degradation, our approach demonstrates a slower and more stable decline, indicating stronger resistance to catastrophic forgetting. The performance gap between our method and the strongest baseline (EFC) becomes increasingly evident as more tasks are introduced, further validating the effectiveness of our class-aware drift compensation mechanism in long sequence scenarios.

Comparison of CADC with SOTA. In the main paper, we modified the proposed drift-guided knowledge distillation (KD) into a vanilla KD, thereby enabling an isolated comparison of the performance between our proposed class-aware drift compensation (CADC) and existing drift compensation approaches. The results presented in Table 4 only report the average incremental accuracy $\mathcal{A}_{inc}$ under the 5/10 task splits across the CIFAR-100 and TinyImageNet datasets. Here, we conduct a more comprehensive evaluation to further validate the superior effectiveness of CADC in comparison with existing methods. The experiments are further conducted on four datasets, including CIFAR-100, TinyImageNet, CUB-200, and ImageNetSubset. Each dataset is uniformly partitioned into 5, 10, or 20 tasks, and the performance is evaluated using two metrics, including the last task accuracy $\mathcal{A}_{last}$ and the average incremental accuracy $\mathcal{A}_{inc}$. CUB-200 is a fine-grained dataset. Following established practices in the literature [2], we adopt a model pre-trained on ImageNet as the initial backbone and perform incremental training on CUB-200. The learning rate across all stages is set to 0.05, with each model trained for 100 epochs. The KD strength is configured to 5, and the batch size is fixed at 64. Similar to other drift compensation approaches, our proposed CADC is also a plug-and-play module. Therefore, all experiments are conducted based on the LwF framework, and the Nearest Class Mean (NCM) classifier is employed to assess the effectiveness of different drift compensation methods in correcting prototype representations. According to the experimental results presented in Tables A2-A3, CADC consistently outperforms existing methods across all three task splits on four datasets. Notably, the performance advantage of CADC becomes more pronounced as the number of task splits increases. For instance, in terms of the $\mathcal{A}_{last}$ on the CUB-200 dataset reported in Table A2, CADC achieves improvements of 0.26%, 1.86%, and 4.19% over the second-best models under the 5/10/20 task split settings, respectively. Similar trends can be observed across other datasets. The superiority of CADC stems from its distinctive capability to model semantic shifts dynamically, which differs fundamentally from vanilla drift compensation approaches. Specifically, CADC achieves class-wise adaptive drift compensation by accounting for varying degrees of semantic shift across different classes, which better aligns with the non-uniform semantic shift in real-world continual learning scenarios.

To comprehensively investigate the trends of model behavior throughout the incremental learning phases, we present line plots in Fig. A2 that illustrate the progression of model capabilities across four datasets. For each dataset, under the 5/10/20 split settings, we uniformly report the trend of the last five tasks to ensure consistency in comparison. It can be observed that, at all displayed learning stages, our proposed CADC method consistently outperforms existing drift compensation approaches. Moreover, the performance gap widens as the number of task splits increases, indicating that semantic shifts during the incremental process are inherently heterogeneous. This finding highlights the necessity of applying class-aware drift compensation to better accommodate varying degrees of feature drift across classes.

Impact of drift compensation weight. The proposed CADC method employs tailored drift compensation weights for each class to facilitate precise prototype correction. An ablation study is conducted here to evaluate the effectiveness of this differentiated compensation strategy. In Eq. (5) of the main paper, we defined the compensation weight τ_i for class $c_i \in \mathcal{C}_{1:t-1}$ as follows:

$$\tau_i = \frac{\epsilon}{1 + \exp\left(-\mathcal{S}^t\left(c_i\right) / \max\left\{\mathcal{S}^t\left(c_i\right)\right\}_{i=1}^{|\mathcal{C}_{1:t-1}|}\right)},$$

where $\mathcal{S}^t\left(c_i\right)$ indicates the modeled semantic shift of class c_i. Then, based on Eq. (6) of the main paper, class-aware drift compensation is performed for the prototype corresponding to class c_i, i.e., the prototype p_i is updated through the following formula:

$$\mathrm{p}_i \leftarrow \Psi\left(\mathrm{p}_i\right) = \mathrm{p}_i + \tau_i \cdot \left(\mathrm{p}_i \cdot \mathbf{W} + \mathbf{b} - \mathrm{p}_i\right).$$

The compensation weight τ is essential to the CADC method. To assess its impact, we set $\tau_i = 1$ for $\forall c_i \in \mathcal{C}_{1:t-1}$ and evaluated the resulting changes in performance. Notably, setting $\tau_i = 1$ applies uniform drift compensation across all classes, causing the CADC method to degrade to a vanilla drift compensation approach. In the experiments, $\tau_i = 1$ is applied solely during prototype updates,

Table A2. Comparison of our class-aware drift compensation method with existing approaches in terms of last task accuracy $\mathcal{A}_{last}$ on four datasets. Base denotes the LwF [5] without any drift compensation for prototypes.

Methods	CIFAR-100			TinyImageNet		
	5 stages	10 stages	20 stages	5 stages	10 stages	20 stages
Base	52.46 ± 0.52	32.78 ± 7.47	34.75 ± 1.01	38.89 ± 0.50	30.73 ± 0.46	24.46 ± 0.67
SDC	53.08 ± 1.25	32.51 ± 7.44	31.99 ± 0.63	43.52 ± 0.66	32.42 ± 0.68	21.97 ± 0.96
LDC	58.54 ± 0.24	45.43 ± 0.30	35.69 ± 0.96	44.41 ± 0.46	36.38 ± 0.65	28.65 ± 0.26
ADC	58.49 ± 0.32	43.62 ± 2.61	33.92 ± 0.46	44.38 ± 0.41	36.46 ± 0.38	28.01 ± 0.93
CADC (Ours)	**58.87 ± 0.17**	**45.96 ± 1.54**	**38.56 ± 0.82**	**45.65 ± 0.47**	**37.81 ± 0.46**	**29.75 ± 0.46**
Methods	CUB-200			ImageNet-Subset		
	5 stages	10 stages	20 stages	5 stages	10 stages	20 stages
Base	43.88 ± 0.81	27.49 ± 0.32	16.56 ± 0.59	58.23 ± 1.17	37.24 ± 1.12	18.26 ± 0.93
SDC	44.97 ± 0.95	29.67 ± 0.34	17.81 ± 0.71	65.23 ± 1.33	47.57 ± 1.64	22.35 ± 1.10
LDC	49.49 ± 0.63	33.50 ± 0.97	17.89 ± 1.21	67.37 ± 0.98	55.31 ± 0.44	42.87 ± 0.39
ADC	51.01 ± 0.70	31.28 ± 1.31	16.83 ± 0.06	65.77 ± 0.79	51.99 ± 0.62	40.78 ± 0.43
CADC (Ours)	**51.27 ± 0.90**	**35.36 ± 1.17**	**22.08 ± 1.23**	**68.17 ± 0.70**	**57.11 ± 1.16**	**45.28 ± 0.60**

Table A3. Comparison of our class-aware drift compensation method with existing approaches in terms of average incremental accuracy $\mathcal{A}_{inc}$ on four datasets. Base denotes the LwF [5] without any drift compensation for prototypes.

Methods	CIFAR-100			TinyImageNet		
	5 stages	10 stages	20 stages	5 stages	10 stages	20 stages
Base	66.97 ± 0.99	57.24 ± 2.20	51.64 ± 1.35	54.71 ± 1.68	46.42 ± 1.60	37.62 ± 0.78
SDC	65.46 ± 0.87	57.31 ± 2.24	49.21 ± 1.04	57.25 ± 1.52	48.46 ± 1.34	35.57 ± 0.38
LDC	71.01 ± 0.88	61.77 ± 1.32	53.54 ± 0.87	57.49 ± 1.60	50.26 ± 1.64	42.65 ± 1.34
ADC	70.92 ± 1.05	61.76 ± 1.42	52.54 ± 0.91	57.38 ± 1.28	50.14 ± 1.38	42.91 ± 1.38
CADC (Ours)	**71.16 ± 0.85**	**62.43 ± 1.47**	**54.87 ± 0.82**	**58.23 ± 1.67**	**51.70 ± 1.83**	**43.67 ± 1.31**
Methods	CUB-200			ImageNet-Subset		
	5 stages	10 stages	20 stages	5 stages	10 stages	20 stages
Base	59.94 ± 0.83	47.90 ± 0.61	36.77 ± 1.11	68.32± 1.05	56.58 ± 1.32	29.13 ± 0.89
SDC	61.60 ± 0.82	50.70 ± 0.69	39.47 ± 1.26	76.35 ± 0.49	65.46 ± 1.55	37.45 ± 0.90
LDC	64.17 ± 0.85	52.16 ± 0.90	38.87 ± 0.95	76.15 ± 0.59	69.51 ± 1.08	56.91 ± 0.58
ADC	64.45 ± 0.70	52.90 ± 1.00	39.60 ± 0.89	76.24 ± 0.58	67.34 ± 1.43	54.81 ± 0.73
CADC (Ours)	**64.86 ± 0.91**	**53.85 ± 0.75**	**42.85 ± 1.17**	**77.19 ± 0.53**	**70.13 ± 0.81**	**57.88 ± 0.54**

while the original value of τ remained effective in drift-guided knowledge distillation, maintaining rigorous control of confounding variables in this ablation study. Experiments are conducted on three datasets, including CIFAR-100, TinyImageNet, and an ImageNet-Subset, with each dataset evenly partitioned into 10 tasks. As reported in Table A4, under two evaluation metrics, setting the compensation weight $\tau_i = 1$ resulted in a substantial reduction in effectiveness across all datasets. For instance, the last task accuracy $\mathcal{A}_{last}$ decreased by 1.15%, 1.66%, and 1.93% on CIFAR-100, TinyImageNet, and the ImageNet subset, respectively. These findings highlight the critical role of class-aware weights in prototype updates for continual learning, given the non-uniform semantic shifts across classes during task evolution.

To systematically assess the gains introduced by class-aware drift compensation compared to a fixed compensation weight $\tau_i = 1$, Fig. A3 illustrates the performance improvement observed across incremental stages on two datasets. As illustrated in Fig. A3, assigning class-aware drift compensation weights yields consistent improvements across most incremental stages, with the gains becoming more evident as tasks progress. This trend arises because uniform compensation under non-uniform semantic shifts leads to inaccurate prototype corrections, causing cumulative deviations from the oracle prototypes over time.

Confusion matrix visualization. Fig. A4 shows the confusion matrices of FCS [4] and our method on CIFAR-100 under a 10-task split. While both methods exhibit a diagonal structure, our method produces a sharper and more concentrated diagonal, indicating more accurate and consistent predictions. In contrast, FCS presents more off-diagonal noise, reflecting frequent misclassifications and higher inter-class confusion. These results highlight the ad-

Table A4. Ablation study on drift compensation weights on CIFAR-100, TinyImageNet, and ImageNet-Subset with 10 tasks.

Settings	CIFAR-100		TinyImageNet		ImageNet-Subset	
	$\mathcal{A}_{last}$	$\mathcal{A}_{inc}$	$\mathcal{A}_{last}$	$\mathcal{A}_{inc}$	$\mathcal{A}_{last}$	$\mathcal{A}_{inc}$
$\tau_i = 1$	46.17 ± 0.83	62.52 ± 1.22	37.22 ± 0.71	51.92 ± 1.77	55.14 ± 0.96	67.86 ± 1.13
$\tau_i =$ Eq.(6)	**47.32 ± 0.56**	**63.18 ± 0.96**	**38.88 ± 0.55**	**52.54 ± 1.82**	**57.07 ± 0.70**	**70.35 ± 0.84**

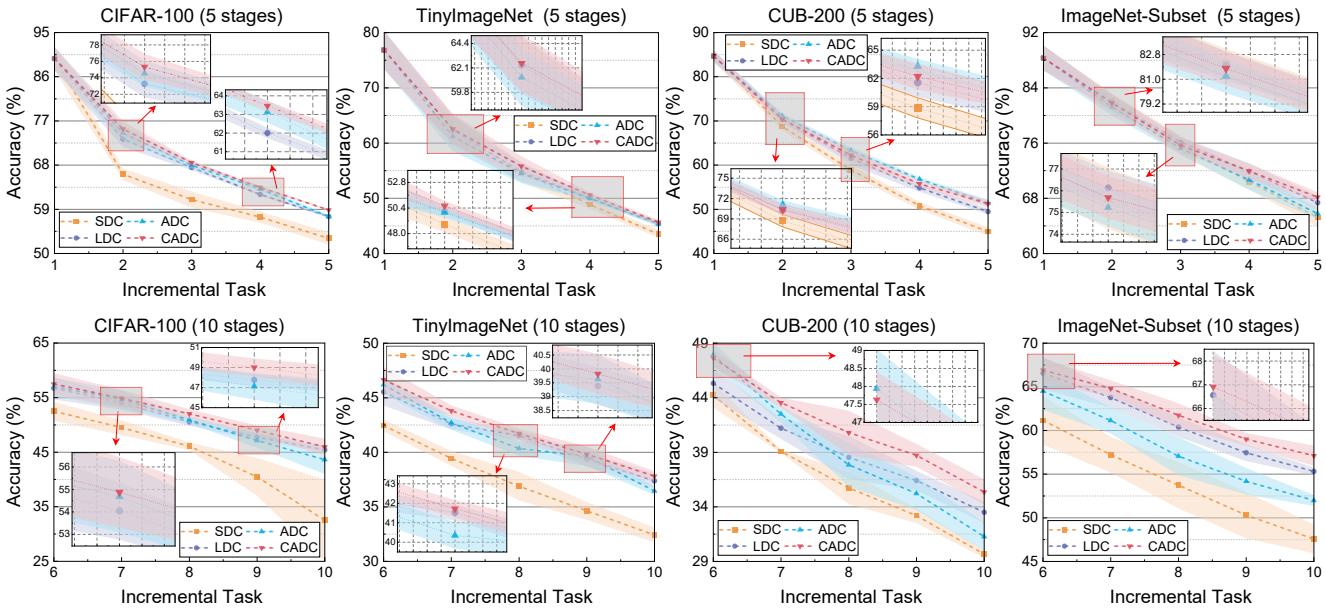

Figure A2. The drift compensation methods are evaluated across four benchmark datasets (CIFAR-100, TinyImageNet, CUB-200, and ImageNet-Subset), with performance comparisons conducted under incremental learning scenarios divided into 5 and 10 stages. The line charts specifically illustrate the performance during the last five phases. CADC is our proposed method.

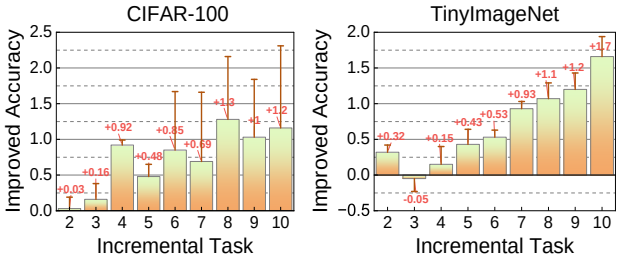

Figure A3. Accuracy improvements from class-aware drift compensation weights via Eq. (6), compared to fixed $\tau_i = 1$, in two datasets, each with 10 tasks.

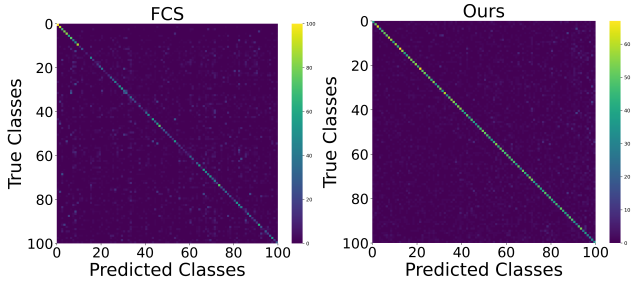

Figure A4. Confusion matrices of two exemplar-free continual learning methods on CIFAR-100 with 10-task split.

vantage of our method in preserving class separability without the use of exemplars. Although FCS also employs a linear layer to estimate drift and update the prototypes of old classes, its effectiveness remains limited. This is mainly because it applies uniform adjustments to all classes, overlooking the fact that semantic shift evolves non-uniformly across different classes as tasks advance. As with other drift compensation methods, it fails to model class-specific prototype correction during the continual learning.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[2] Dipam Goswami, Albin Soutif-Cormerais, Yuyang Liu, Sandesh Kamath, Bartłomiej Twardowski, and Joost Van

De Weijer. Resurrecting old classes with new data for exemplar-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 28525–28534, 2024.

[3] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[4] Qiwei Li, Yuxin Peng, and Jiahuan Zhou. Fcs: Feature calibration and separation for non-exemplar class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 28495–28504, 2024.

[5] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018.

[6] Simone Magistri, Tomaso Trinci, Albin Soutif, Joost van de Weijer, and Andrew D. Bagdanov. Elastic feature consolidation for cold start exemplar-free incremental learning. In *International Conference on Learning Representations (ICLR)*, 2024.

[7] Simone Magistri, Tomaso Trinci, Albin Soutif-Cormerais, Joost van de Weijer, and Andrew D. Bagdanov. Efc++: Elastic feature consolidation with prototype re-balancing for cold start exemplar-free incremental learning. *CoRR*, 2025.

[8] Lu Yu, Bartłomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6980–6989, 2020.

[9] Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, and De-Chuan Zhan. Pycil: a python toolbox for class-incremental learning. *SCIENCE CHINA Information Sciences*, 66(9):197101, 2023.