Adapter Merging with Centroid Prototype Mapping for Scalable Class-Incremental Learning

Takuma Fukuda[1] Hiroshi Kera[1,2] Kazuhiko Kawamoto[1]

[1] Chiba Univ. [2] Zuse Institute Berlin

CVPR Nashville JUNE 11-15, 2025

CHIBA UNIVERSITY

Class-Incremental Learning (CIL)

CIL with pretrained model

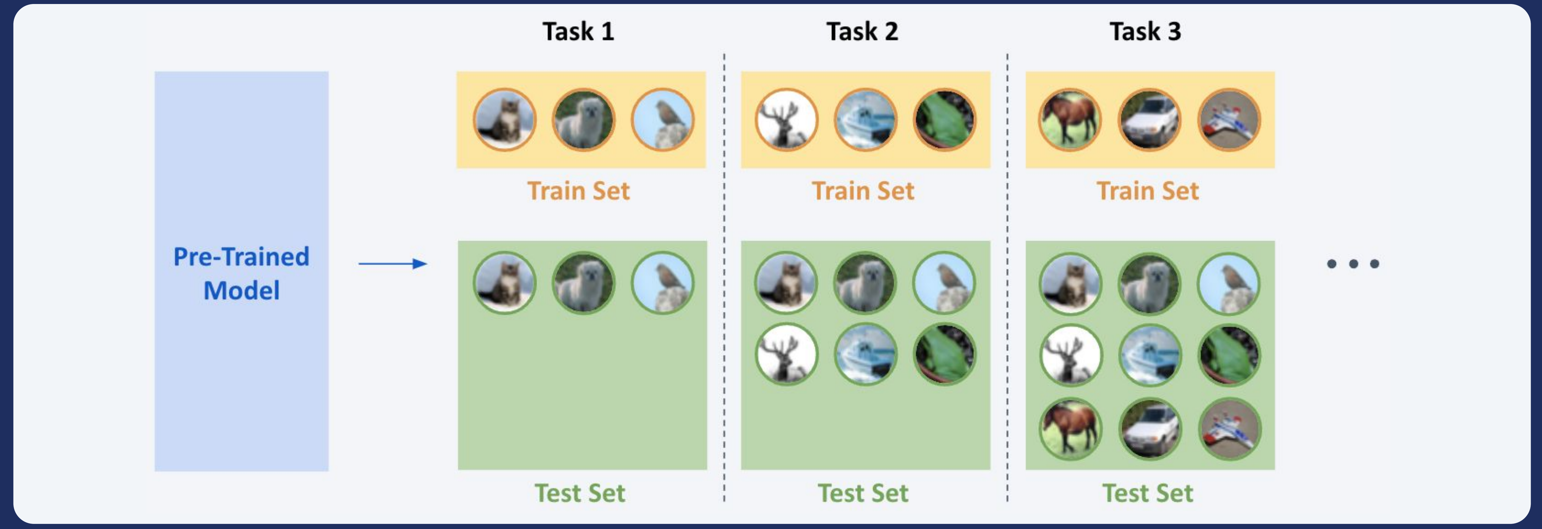
Incrementally adapt a pretrained model to tasks composed of new classes, while preventing **catastrophic forgetting**

Catastrophic forgetting

Forget previously learned knowledge when adapting to new tasks, degrading performance on old classes

Exemplar-free CIL

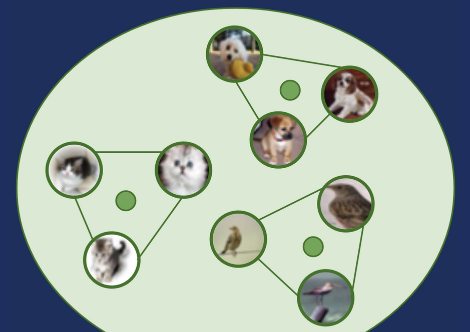
Does not use **exemplars** — representative samples from previous classes — which are often restricted by privacy concerns



Prototype-based CIL

Classify samples by comparing their features to **prototypes**, which are the average feature vector of each class

In CIL, prototypes are calculated for each task and stored during subsequent task

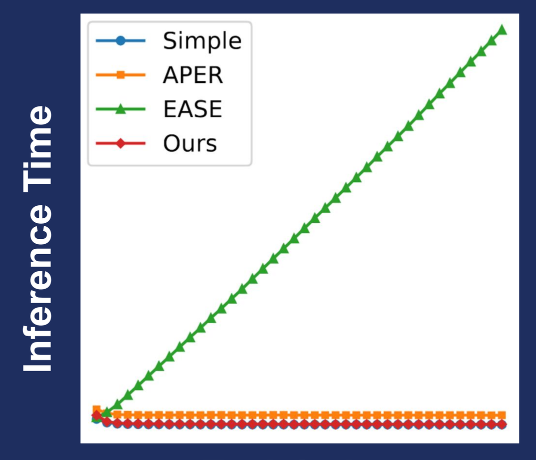


Motivation

Inference Scalability

In real-world applications requiring long-term deployment, scalability issue becomes a critical

However, there is a trade-off between scalability and accuracy.

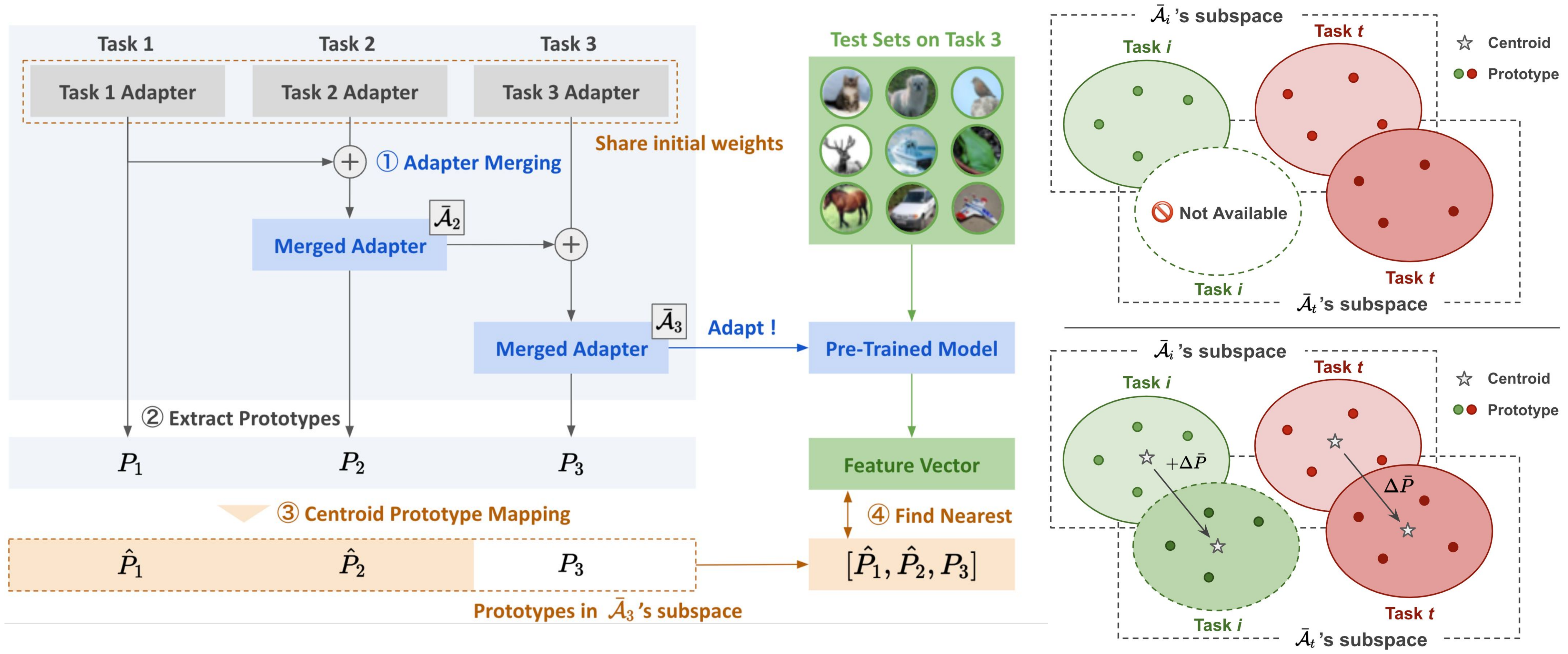


Contributions

Achieved accuracy comparable to SoTA methods while maintaining inference time similar to the fastest approaches

- ➢ Merge task-specific adapters to achieve scalability
- ➢ Adopt prototype alignment to resolve merging inconsistencies

Method – ACMap



Adapter Merging

Weight Averaging [3]

- ➢ Train adapters with shared initial weights
- ➢ Average adapter weights iteratively

$$\bar{\theta}_t = \left(1 - \frac{1}{t}\right)\bar{\theta}_{t-1} + \frac{1}{t}\theta_t$$

Initial Weight Replacement (IR)

- ➢ Replaces initial weight θ_{init} with the weight θ_1 learned from task 1
  - ○ Encourages the formation of a low-loss basin

Early Stopping Strategy

- ➢ Stops adapter merging at task L
  - ○ As tasks increase, θ_{t-1} and θ_t becomes nearly identical

Centroid Prototype Mapping (CM)

Notations

$\bar{A}_i$: merged adapter in task i, $P_i(\bar{A})$: prototypes with $\bar{A}$ in task i

Prototype Distribution Shift

- ➢ Previous prototypes $P_i(\bar{A}_t)$ cannot be calculated because their data are not available in task t
- ➢ They also cannot be directly reused, as prototype distribution shift occurs between $P_i(\bar{A}_i)$ and $P_i(\bar{A}_t)$

Centroid Prototype Mapping

- ➢ Shift Previous prototype $P_i(\bar{A}_i)$ with the centroid shift ΔP from $P_t(\bar{A}_i)$ to $P_t(\bar{A}_t)$

$$\Delta P = \mathbb{E}[P_t(\bar{A}_t) - P_t(\bar{A}_i)]$$

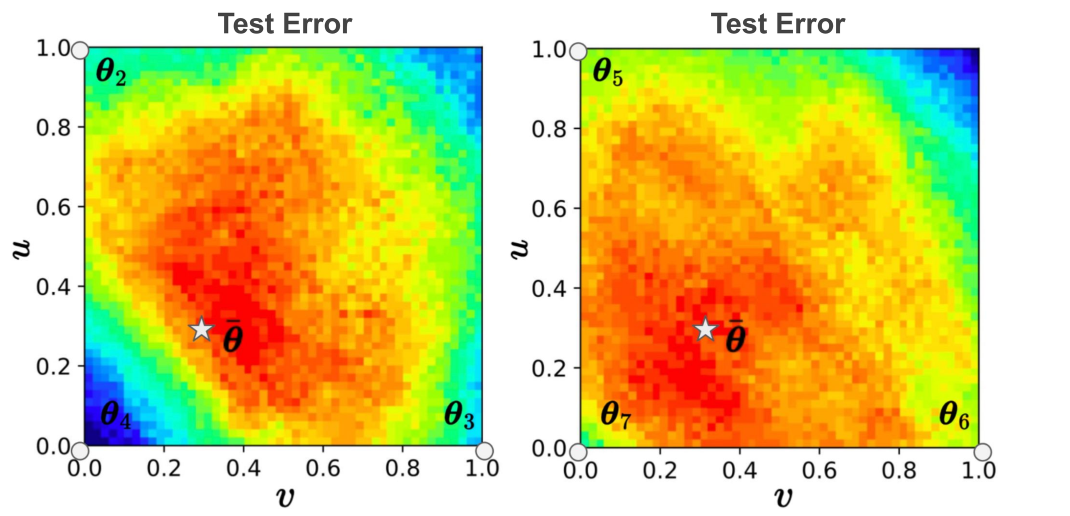
$$P_i(\bar{A}_t) \approx P_i(\bar{A}_i) + \Delta P$$

Landscape Analysis for Adapters

- ➢ Loss landscape of three adapters through the linear interpolation

$$\theta = u\theta_{t-1} + v\theta_t + (1-u-v)\theta_{t+1}, \ (0 \le u, v \le 1)$$

- ➢ Averaged weight $\bar{\theta}_t$ is located in low-loss basin (red region)



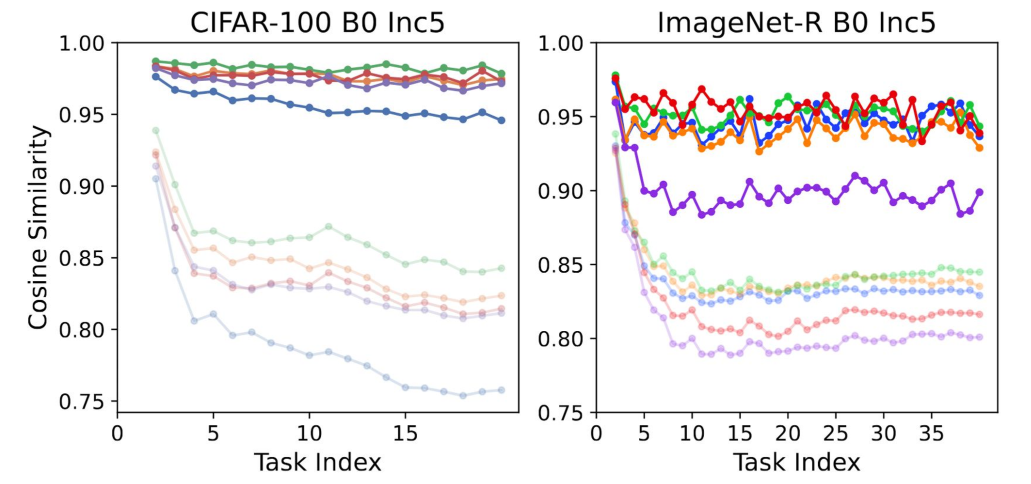
Effectiveness of Prototype Mapping

- ➢ Solid lines indicate effectiveness of prototype mapping

$$\mathrm{Sim}(P_1(\bar{A}_1), P_1(\bar{A}_t))$$

- ➢ Semi-transparent lines indicate prototype distribution shift

$$\mathrm{Sim}(\hat{P}_1(\bar{A}_1), P_1(\bar{A}_t))$$

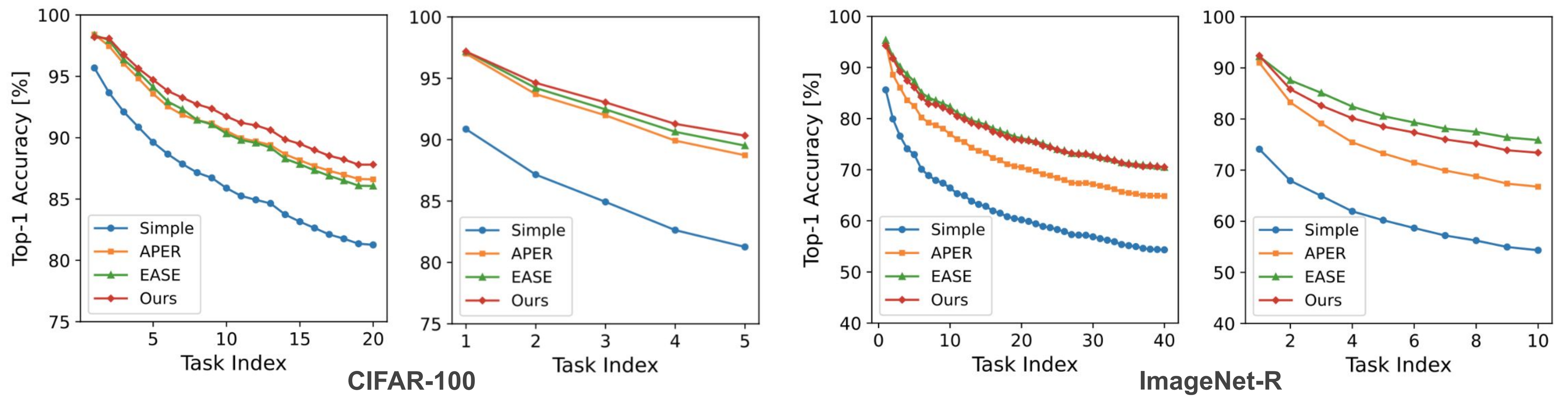


Results

$\bar{A}$: Top-1 accuracy averaged across all tasks B-n : Number of classes learned initially

A_T : Top-1 accuracy on the final task Inc-m : Number of new classes added in each task

Method	CIFAR B0 Inc5 $\bar{A}$	CIFAR B0 Inc5 A_T	CUB B0 Inc10 $\bar{A}$	CUB B0 Inc10 A_T	IN-R B0 Inc5 $\bar{A}$	IN-R B0 Inc5 A_T	IN-A B0 Inc20 $\bar{A}$	IN-A B0 Inc20 A_T	VTAB B0 Inc10 $\bar{A}$	VTAB B0 Inc10 A_T
SimpleCIL [1]	87.57	81.26	92.20	86.73	62.58	54.55	59.77	48.91	85.99	84.38
APER [1]	90.65	85.15	92.21	86.73	72.35	64.33	60.47	49.37	85.95	84.35
EASE [2]	91.51	85.80	**92.23**	86.81	**78.31**	**70.58**	**65.34**	55.04	**93.61**	**93.55**
Ours w/o IR	91.54	87.35	91.74	**86.96**	76.56	70.08	64.00	54.67	90.28	86.25
Ours	**92.04**	**87.81**	91.56	86.66	77.31	70.49	65.19	**56.19**	91.21	87.56



Inference Time Comparison

Method	Time (s)	Time Ratio	Complexity
SimpleCIL	22.6	×0.96	$\mathcal{O}(1)$
APER	44.1	×1.88	$\mathcal{O}(1)$
EASE	916.5	×39.0	$\mathcal{O}(T)$
ACMap (Ours)	23.5	-	$\mathcal{O}(1)$

On task 40 of ImageNet-R:

Compared to SimpleCIL and APER, ACMap achieves comparable speed but higher accuracy

Compared to EASE, ACMap achieves comparable accuracy but is 40x faster

Ablation Study

IR	CM	CIFAR B0 Inc5 $\bar{A}$	CIFAR B0 Inc5 A_T	IN-R B0 Inc5 $\bar{A}$	IN-R B0 Inc5 A_T
		90.46	86.32	75.99	69.55
	✓	91.53	87.35	76.47	69.88
✓		91.07	86.85	76.56	69.80
✓	✓	**92.01**	**87.73**	**77.10**	**70.25**

Threshold	CIFAR B0 Inc5 $\bar{A}$	CIFAR B0 Inc5 A_T	IN-R B0 Inc5 $\bar{A}$	IN-R B0 Inc5 A_T
$L = 0$	91.07	86.85	76.56	69.80
$L = 5$	91.88	87.61	76.81	69.87
$L = 10$	92.00	87.76	77.09	70.25
$L = 20$	**92.04**	**87.80**	**77.27**	**70.37**

Future Work

ACMap currently maintains a single merged adapter to ensure scalability. However, this design may limit performance, especially when tasks differ significantly in domain.

A potential extension is to maintain multiple merged adapters and dynamically select the most appropriate one for each task.

[1] Revisiting Class-Incremental Learning with Pre-Trained Models: Generalizability and Adaptivity are All You Need. IJCV, 2024
[2] Expandable Subspace Ensemble for Pre-Trained Model-Based Class-Incremental Learning. CVPR, 2024
[3] Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. ICML, 2022