CAMP: COUPLED ALIGNMENT OF MANIFOLD AND PROTOTYPES FOR EXEMPLAR-FREE CLASS-INCREMENTAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Exemplar-free class-incremental learning (EFCIL) requires sequentially learning new classes without storing past data, a setting where catastrophic forgetting stems from the progressive drift of feature representations across tasks. Existing methods treat feature drift and classifier misalignment as separate concerns (distilling the backbone, post-hoc correcting prototypes) without a unified account of how the two interact. We show they are two sides of one geometric problem: the classifier induces a pullback metric on the feature manifold, so manifold stability is a prerequisite for valid metric recalibration, and the two must be coupled rather than staged. We propose CAMP (Coupled Alignment of Manifold and Prototypes), a framework where these two processes are interlocked as a causal pipeline rather than stacked as independent modules. CAMP combines Feature Manifold Constraint (FMC), a gradient-complete training constraint, with Kernel Shift Compensation (KSC), a non-parametric minimax-optimal estimator of residual drift, and Prototype-Anchored Classifier Recalibration (PACR), which re-anchors drifted weights to corrected prototypes, forming one causal chain. Experiments on multiple benchmarks show state-of-the-art accuracy and superior retention.

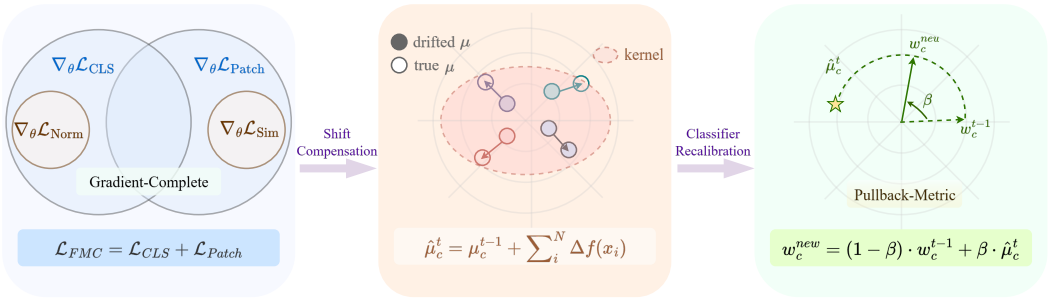
1 INTRODUCTION

Class-incremental learning (CIL) trains models to sequentially acquire new class knowledge while retaining what was previously learned (McCloskey & Cohen, 1989; Li & Hoiem, 2017; He et al., 2025a). This capability is essential for applications where models must adapt to evolving data without full retraining. The central obstacle is catastrophic forgetting: learning new tasks interferes with old knowledge, reflecting the fundamental tension between stability and plasticity (Parisi et al., 2019). The exemplar-free setting (EFCIL) further complicates this picture: privacy regulations and storage constraints often prohibit retaining old-class samples, leaving the model with no direct access to old data. Each new task therefore shifts the model's internal representations, and this semantic drift lies at the heart of catastrophic forgetting in EFCIL.

Semantic drift in EFCIL manifests as two distinct but related problems: feature drift in the backbone and classifier misalignment in the decision space. Feature drift occurs when backbone updates on new tasks displace old-class representations from their previously learned positions. To counter this, existing methods combine knowledge distillation to regularize the backbone (Li & Hoiem, 2017; Li et al., 2024) with post-hoc compensation (Gomez-Villa et al., 2024; Rao et al., 2026; Yu et al., 2020) that estimates how far old-class prototypes have moved. Classifier misalignment arises because decision boundaries optimized on new classes grow biased away from old-class regions. The standard remedy stores class prototypes and covariances to generate Gaussian pseudo-samples around old-class centroids, and retrains the classifier via Classifier Alignment (CA) (Zhang et al., 2023; Wu et al., 2025). These strategies reduce drift to varying degrees and thereby alleviate forgetting.

However, we identify two specific shortcomings in how existing methods tackle these two problems. Firstly, existing methods apply knowledge distillation to limit feature drift, but select distillation targets heuristically. They combine distillation targets from different perspectives (e.g., CLS tokens (He et al., 2025a), patch tokens (Wu et al., 2025), feature norms (Rao et al., 2026), and inter-patch similarity (Tung & Mori, 2019)) without a principled understanding of which are necessary and which are redundant. Secondly, the standard Classifier Alignment (CA) protocol treats

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.



(a) Feature Manifold Constraint (b) Kernel Shift Compensation (c) Prototype-Anchored Classifier Recalibration

Figure 1: Overview of CAMP. (a) FMC: during backbone training, gradient-complete CLS and patch token distillation preserves the feature manifold. (b) KSC: after training, the Epanechnikov-kernel estimator corrects residual prototype drift via non-parametric local compensation. (c) PACR: corrected prototypes re-anchor the drifted classifier weights through a single pullback-metric interpolation step.

the classifier as a set of decision boundaries and re-trains it from scratch using Gaussian pseudo-samples drawn around old-class centroids. This overlooks the pullback metric structure that the linear classifier induces on the feature manifold (Chaudhry, 2026; Papyan et al., 2020). Re-learning from scratch discards the metric structure encoded in the classifier weights. These two shortcomings are in fact two sides of the same geometric problem. The pullback metric perspective reveals that classifier validity depends on manifold stability: correcting the metric without first stabilizing the manifold is counterproductive, because the prototypes used as anchors have themselves drifted from their true positions. The two must therefore be addressed as a single coupled system rather than as independent stages.

We propose Coupled Alignment of Manifold and Prototypes (CAMP), in which manifold preservation and classifier recalibration are interlocked as a causal pipeline rather than stacked as independent modules (Figure 1). CAMP has three stages. Feature Manifold Constraint (FMC) is a gradient-completeness analysis that identifies the minimal non-redundant set of distillation objectives. Kernel Shift Compensation (KSC) is a non-parametric Epanechnikov-kernel (Epanechnikov, 1969) estimator, provably minimax optimal among non-negative compactly supported kernels, that corrects residual prototype drift without any training. Prototype-Anchored Classifier Recalibration (PACR) exploits the pullback metric induced by the classifier to re-anchor drifted weights to the corrected prototypes in a single interpolation step. Our contributions are summarized as follows:

- We propose CAMP, a framework where manifold preservation and classifier recalibration are interlocked as a causal pipeline rather than stacked as independent modules.
- We analyze the gradients of the candidate distillation losses and show that CLS and patch distillation jointly form the minimal sufficient basis for manifold preservation, and on this basis propose the Feature Manifold Constraint (FMC) loss.
- We introduce Kernel Shift Compensation (KSC), a non-parametric Epanechnikov-kernel drift estimator with minimax optimality and compact support, and propose Prototype-Anchored Classifier Recalibration (PACR), a single-step pullback-metric re-anchoring that corrects classifier weights without re-training.
- Extensive experiments on multiple benchmarks demonstrate that CAMP achieves state-of-the-art performance with superior retention, and confirm the causal dependency among its components.

2 Related Work

2.1 Exemplar-Free Class-Incremental Learning

EFCIL requires sequentially learning new classes without storing past samples, making catastrophic forgetting particularly severe. Traditional approaches fall into three categories: regularization-based methods (Li & Hoiem, 2017) constrain parameter updates via distillation, expansion-based meth-

ods (Hung et al., 2019; Li et al., 2019) grow the architecture per task, and memory-based methods (Aljundi et al., 2019; Chrysakis & Moens, 2020) replay stored exemplars (the latter unavailable here).

With pre-trained Vision Transformers (ViTs) (Dosovitskiy, 2020), the field has shifted toward parameter-efficient fine-tuning (PEFT), which adapts only a small set of parameters while keeping the backbone frozen. PEFT-based continual learning methods fall into three categories. Prompt-based approaches, such as L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a) and CODA-Prompt (Smith et al., 2023), learn task-specific prompts inserted into the input sequence, adapter-based methods (Gao et al., 2023; Zhou et al., 2025; 2024) insert lightweight modules between transformer layers, and LoRA-based methods (Hu et al., 2022) inject low-rank updates into attention weights, offering parameter efficiency and natural task isolation. CL-LoRA (He et al., 2025a) allocates separate LoRA branches per task, while O-LoRA (Wang et al., 2023) and InfLoRA (Liang & Li, 2024) enforce orthogonality constraints to reduce interference, and PLAN (Wang et al., 2025) and SABER (Luo et al., 2026) use orthonormal initialization for task-specific subspaces. CAMP follows the LoRA-based paradigm, updating only low-rank adapters and the classifier, with all adapters merged into the backbone at inference time at no additional cost.

2.2 Semantic Drift in EFCIL

Semantic drift, the progressive shift of old-class representations caused by backbone updates, is the root cause of forgetting in EFCIL, and existing methods address it through compensation and recalibration. For prototype compensation, the approaches differ mainly in how the drift field is modeled: SLDC (Rao et al., 2026) fits a weak-linear transformation (efficient but rigid), FCS (Li et al., 2024) uses neural networks or optimal transport (flexible but over-parameterized under scarce per-class samples), and LDC (Gomez-Villa et al., 2024) trains an adapter to absorb the shift (at the cost of inference overhead). A second family avoids explicit mappings altogether: SDC (Yu et al., 2020), SSIAT (Tan et al., 2024) and MACIL (Wu et al., 2025) use non-parametric Gaussian-kernel estimators, AdaGauss (Rypeść et al., 2024) adapts class covariances against task-recency bias, and DPCR (He et al., 2025b) estimates the shift by dual projection followed by classifier reconstruction. None of these justifies the kernel choice theoretically. MACIL additionally distils patch tokens with a cosine-similarity weighting. As our gradient-subspace analysis shows (Section 3.2), that similarity term is a strict gradient subset of the patch-position term, so it adds a redundant objective rather than a stronger constraint, a consequence of selecting distillation targets without a completeness criterion. For classifier recalibration, Classifier Alignment (CA) (Zhang et al., 2023; Wu et al., 2025) stores prototypes and covariances to generate Gaussian pseudo-samples and re-trains the classifier. Two further works are close in spirit: Elastic Feature Consolidation (Magistri et al., 2024) couples a feature-drift regularizer with a feature-space pseudo-metric, prototype updates and classifier adaptation, and (Zhai et al., 2024) couples selective patch-level distillation with prototype restoration. Like the methods above, both assume a backbone trained from scratch, so we position against them conceptually rather than empirically. Critically, all of these treat compensation and recalibration as independent post-processing stages, overlooking their geometric coupling. In contrast, CAMP couples a gradient-complete manifold constraint with pullback-metric-based recalibration, linked by the causal insight that the manifold must be preserved before the metric can be corrected.

3 Methodologies

3.1 Preliminaries

3.1.1 Class-Incremental Learning.

We consider a model trained sequentially on T tasks, each introducing a disjoint class set $\mathcal{C}_t$ and providing data $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{N_t}$ with no exemplars from preceding classes $\mathcal{C}_{0:t-1}$. During training on task t only $\mathcal{D}_t$ and the previous model $f_{\theta_{t-1}}$ are accessible, while at evaluation all seen classes $\mathcal{C}_{0:t}$ must be classified. Following existing EFCIL methods (Rao et al., 2026; Wu et al., 2025), we assume a pre-trained Vision Transformer (ViT) (Dosovitskiy, 2020) backbone with a linear classifier $g(\cdot)$ on top, trained with cross-entropy:

$$\mathcal{L}_{\text{CE}} = -\mathbb{E}_{(x,y)\sim\mathcal{D}_t}\left[\log \frac{\exp(g(f_\theta(x))_y)}{\sum_{j\in\mathcal{C}_t}\exp(g(f_\theta(x))_j)}\right]. \tag{1}$$

3.1.2 LOW-RANK ADAPTATION.

LoRA (Hu et al., 2022) parameterizes an update on a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times d}$ by a down-projection $A \in \mathbb{R}^{r\times d}$ and an up-projection $B \in \mathbb{R}^{d\times r}$ with rank $r \ll d$, giving the forward pass

$$h = W_0 x + BAx. \tag{2}$$

In the continual setting we keep a task-specific pair $\{A_t, B_t\}$ per task, updating only the current pair while W_0 and all previous pairs stay frozen, and merge everything at test time as

$$W = W_0 + \sum_{i=1}^{T} B_i A_i, \tag{3}$$

which requiring zero additional inference cost.

3.2 FMC: FEATURE MANIFOLD CONSTRAINT

We use feature manifold operationally to mean the image of the backbone, $\mathcal{M} = f_\theta(\mathcal{X}) \subset \mathbb{R}^d$, with the Euclidean metric inherited from feature space. Three quantities track its change across tasks: the CLS representation c, the patch tokens $\{p_i\}$, and the class prototypes μ_c. Catastrophic forgetting in EFCIL occurs when backbone updates deform this manifold, so that old-class representations drift from their previously learned positions. Knowledge distillation (KD) naturally serves the purpose of constraining the manifold: matching features between f_{θ_t} and $f_{\theta_{t-1}}$ penalizes backbone deviation. Existing KD methods working on backbone have adopted various distillation targets, such as CLS token (He et al., 2025a), patch tokens (Yang et al., 2024), feature norms (Rao et al., 2026), and inter-patch similarity matrices (Tung & Mori, 2019). While each captures a partial aspect of manifold geometry, no prior work has systematically analyzed whether these constraints are independent or complete. This raises a fundamental question: which combination of distillation objectives constitutes a complete and non-redundant basis for constraining the entire feature manifold? We answer this question through a gradient subspace analysis.

3.2.1 GRADIENT SUBSPACE ANALYSIS

Our key observation is that a ViT produces exactly two types of feature-level outputs: the CLS token $c \in \mathbb{R}^d$ (an aggregated global representation) and N patch tokens $\{p_i\}_{i=1}^N \subset \mathbb{R}^d$ (local spatial representations). The natural derived quantities from these outputs are also limited: the scalar magnitude of the CLS token ($\|c\|$) and the pairwise cosine similarities between patch tokens ($p_i \cdot p_j / \|p_i\|\|p_j\|$). This exhausts the space of feature-level quantities one can distill in a ViT. We therefore analyze four distillation losses that jointly span this space:

$$\mathcal{L}_{\text{CLS}} = \|c^{(t)} - c^{(t-1)}\|^2, \tag{4}$$

$$\mathcal{L}_{\text{Patch}} = \sum\nolimits_i \|p_i^t - p_i^{t-1}\|^2, \tag{5}$$

$$\mathcal{L}_{\text{Norm}} = (\|c^{(t)}\| - \|c^{(t-1)}\|)^2, \tag{6}$$

$$\mathcal{L}_{\text{Sim}} = \|M^t - M^{(t-1)}\|_F^2, \;\; M_{ij} = \frac{p_i \cdot p_j}{\|p_i\|\|p_j\|}, \tag{7}$$

Here $c^{(t)}, c^{(t-1)} \in \mathbb{R}^d$ denote the CLS tokens at tasks t and $t-1$, and $p_i^t, p_i^{t-1} \in \mathbb{R}^d$ the i-th patch token ($i = 1, \ldots, N$). The CLS token decomposes as an attention-weighted sum of patch tokens, $c = \sum_i w_i p_i$, with weights w_i from the final self-attention layer, and $M \in \mathbb{R}^{N\times N}$ is the patch-pair cosine similarity matrix. We write $\|\cdot\|$ for the L_2 norm and $\|\cdot\|_F$ for the Frobenius norm.

We decompose $\nabla_\theta \mathcal{L}$ for each loss and identify the independent gradient components each contributes. This gradient-centric perspective is what makes our analysis exhaustive: any feature-level constraint ultimately acts on θ through its gradient, so comparing gradient subspaces tells us exactly which losses are independent and which are subsets of others. Thus, the analysis establishes four results that collectively identify the minimal sufficient distillation set:

(i) Norm adds no direction beyond CLS. Both losses depend on θ only through c, so it suffices to compare the feature-space directions they probe:

$$\frac{\partial \mathcal{L}_{\text{Norm}}}{\partial c} = \frac{2(\|c\| - \|c^{(t-1)}\|)}{\|c\|}, \qquad \frac{\partial \mathcal{L}_{\text{CLS}}}{\partial c} = 2\,(c - c^{(t-1)}). \tag{8}$$

Table 1: Gradient subspace coverage of the four distillation losses. ✓ means contributes gradient and "–" means no contribution. ($\subset X$) means strict subset of X's gradient span.

Gradient	CLS	Patch	Norm	Sim
Token position	✓	✓	–	–
Token magnitude	–	✓	–	–
Token direction	–	✓	–	✓($\subset$ Patch)
Attention	✓	–	–	–
Global magnitude	✓	–	✓($\subset$ CLS)	–

The norm gradient is always a scalar multiple of c, acting only along the radial direction, whereas the CLS gradient sweeps all of $\mathbb{R}^d$ as the target $c^{(t-1)}$ varies over the data. Norm regularization therefore contributes no independent constraint.

(ii) $\mathcal{L}_{\text{Sim}}$ is a strict gradient subset of $\mathcal{L}_{\text{Patch}}$, because $\mathcal{L}_{\text{Sim}}$ depends only on patch directions (the L_2 normalization in M_{ij} discards magnitude), whereas $\mathcal{L}_{\text{Patch}}$ constrains position, magnitude and direction jointly.

(iii) Only $\mathcal{L}_{\text{CLS}}$ constrains attention aggregation, through the term $\sum_i p_i \, (\partial \mathcal{L}_{\text{CLS}}/\partial c)(\partial w_i/\partial \theta)$: without it, attention drift alters the global representation $c = \sum_i w_i p_i$ even when patch positions are perfectly preserved.

(iv) Consequently $\nabla_\theta \mathcal{L}_{\text{CLS}} \cup \nabla_\theta \mathcal{L}_{\text{Patch}}$ spans the full distillation gradient subspace with zero redundancy, whereas $\mathcal{L}_{\text{Norm}}$ and $\mathcal{L}_{\text{Sim}}$ add no new gradient dimensions and can cause over-constraint. The minimal sufficient constraint is exactly $\mathcal{L}_{\text{CLS}} + \mathcal{L}_{\text{Patch}}$.

Table 1 summarizes the results. The complete derivations of (i)–(iv) are given in Appendix B.

3.2.2 FMC Loss

The FMC loss, which distills CLS and Patch tokens, is defined as follows:

$$\mathcal{L}_{\text{FMC}} = \sum_i \|c_i^{(t)} - c_i^{(t-1)}\|^2 + \gamma \sum_{i,j} \|p_{i,j}^{(t)} - p_{i,j}^{(t-1)}\|^2, \tag{9}$$

where γ is the hyperparameter that balances the CLS and patch constraints. The two terms are complementary: the CLS term anchors the global position of each point, while the patch term preserves the local structure among patch embeddings (their relative positions and pairwise relations). Because $c = \sum_i w_i p_i$, patch drift propagates into the CLS representation and hence into the prototypes that KSC corrects and PACR re-anchors. Appendix C makes this transfer explicit. Together the two terms ensure the feature manifold remains approximately invariant across tasks, which is the geometric prerequisite for the downstream coordinate and metric corrections.

3.3 KSC: Kernel Shift Compensation

Though FMC keeps the manifold approximately invariant, a residual coordinate drift remains. To correct it, we first define the prototype of class c after task t as the mean feature representation:

$$\mu_c^{(t)} = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{1}[y_i = c] \, f_{\theta_t}(x_i), \tag{10}$$

where N_c is the number of samples of class c. The class covariance is defined analogously:

$$\Sigma_c^{(t)} = \frac{1}{N_c - 1} \sum_{i:y_i=c} \big(f_{\theta_t}(x_i) - \mu_c^{(t)}\big)\big(f_{\theta_t}(x_i) - \mu_c^{(t)}\big)^\top, \tag{11}$$

and is used for pseudo-sample generation in the CA stage. We do not compensate the covariance shift (FMC already limits its magnitude, and with few samples per class the estimate is too noisy to correct reliably), so correcting the mean alone suffices. Following (Yu et al., 2020), we approximate the unknown prototype shift using the embedding shift of current-task data, since both stem from the same backbone update. Specifically, for each sample x_i we compute the embedding shift:

$$\Delta f(x_i) = f_{\theta_t}(x_i) - f_{\theta_{t-1}}(x_i), \tag{12}$$

and estimate each old prototype's shift as a kernel-weighted average of these embedding shifts. We use the Epanechnikov (Epanechnikov, 1969) kernel rather than the customary Gaussian kernel, for its compact support (samples beyond a calibrated distance contribute exactly zero) together with

its minimax optimality among non-negative compactly supported kernels. Appendix D derives this optimality, states the assumptions under which it holds, and analyses the kernel's behavior in high dimensions and under an adaptive bandwidth. The KSC estimator is:

$$\hat{\mu}_c^{(t)} = \mu_c^{(t-1)} + \frac{\sum_i K\left(\frac{\|f_{\theta_{t-1}}(x_i) - \mu_c^{(t-1)}\|}{\sigma}\right) \cdot \Delta f(x_i)}{\sum_i K\left(\frac{\|f_{\theta_{t-1}}(x_i) - \mu_c^{(t-1)}\|}{\sigma}\right)}, \tag{13}$$

where σ is the median distance from current-task samples to the old prototype $\mu_c^{(t-1)}$, computed per class without any learnable parameters, and

$$K(u) = \frac{3}{4}(1 - u^2) \cdot \mathbf{1}_{|u| \leq 1}, \tag{14}$$

where $u = \|f_{\theta_{t-1}}(x_i) - \mu_c^{(t-1)}\|/\sigma$ and $\mathbf{1}_{|u|\leq 1}$ is the indicator function. KSC is non-parametric and requires no training, making it both computationally efficient and robust to limited data.

3.4 PACR: Prototype-Anchored Classifier Recalibration

FMC and KSC together take care of the backbone side of forgetting: the manifold is preserved and the prototypes are realigned, so the representation itself no longer drifts. The classifier, however, is left unaddressed, and accumulates a bias of its own against older classes. The standard remedy is Classifier Alignment (CA): sample Gaussian pseudo-features around each old-class prototype and fine-tune the classifier with cross-entropy loss (Zhang et al., 2023; Wu et al., 2025). However, CA treats the classifier merely as a set of decision boundaries to be re-trained, overlooking a key geometric fact: the classifier induces the pullback metric $g_W(u, v) = u^\top (WW^\top)v$ on the feature manifold (Chaudhry, 2026), in which each row w_c is the differential of the class-c score and measures how strongly a point belongs to class c. Re-learning from scratch discards the metric structure already encoded in W and trains on pseudo-samples that lie off the true manifold. Rather than re-learning, we leverage the pullback metric perspective to directly re-anchor the drifted classifier to the KSC-corrected prototypes $\hat{\mu}_c^{(t)}$.

3.4.1 Pullback Metric Re-anchoring

PACR instead re-anchors the drifted classifier to the corrected prototype, acting as an initialization for the CA stage that follows:

$$w_c^{\text{new}} = (1 - \beta) \cdot w_c^{t-1} + \beta \cdot \frac{\hat{\mu}_c^{(t)}}{\|\hat{\mu}_c^{(t)}\|}, \tag{15}$$

where $\hat{\mu}_c^{(t)}$ is the KSC-corrected prototype and $\beta \in [0, 1]$ controls the re-anchoring strength. PACR is applied once, immediately before the CA stage, and only to old classes. New classes are naturally aligned to their prototypes during training and need no recalibration. It therefore sets the starting value of each old-class weight, after which CA proceeds exactly as in the standard protocol, with no parameters frozen. The benefit is therefore not that the re-anchoring survives CA unchanged, but that CA initialized from corrected prototypes converges to a solution closer to the true class geometry than CA initialized from fully drifted weights. Geometrically, PACR pulls the drifted classifier toward the corrected prototype direction, which remains a reliable anchor thanks to FMC. Setting $\beta = 0$ disables PACR entirely and reduces the pipeline to plain CA, which is the reference point we use when interpreting the ablation. A theoretical justification is provided in Appendix E.

PACR is therefore not a standalone module: its re-anchoring requires a reliable anchor $\hat{\mu}_c^{(t)}$, which in turn depends on accurate coordinate compensation (KSC) and a preserved manifold (FMC). If FMC is removed, the manifold deforms and prototypes drift far from their true positions. KSC can only partially correct this, and re-anchoring to a still-mispositioned prototype pulls w_c in the wrong direction. The three components thus form a causal pipeline rather than an arbitrary collection of techniques. The total loss of backbone training is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{FMC}}. \tag{16}$$

The pipeline is summarized in Appendix A (Algorithm 1).

Table 2: Comparison with EFCIL methods on ImageNet-R and CUB-200. Best and second-best results are in **bold** and underline, respectively. All reported values are averaged over three random seeds and ± denotes the standard deviation. Per-task accuracy curves are provided in Appendix F.

Method	IN-R (T=10)			IN-R (T=20)			CUB (T=10)		
	Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓
L2P	$70.85_{\pm 0.45}$	$76.01_{\pm 0.28}$	$4.07_{\pm 0.45}$	$68.14_{\pm 0.46}$	$73.56_{\pm 0.73}$	$4.89_{\pm 0.54}$	$68.10_{\pm 0.54}$	$77.63_{\pm 0.61}$	$12.01_{\pm 2.12}$
DualPrompt	$65.58_{\pm 0.93}$	$71.32_{\pm 0.50}$	$4.08_{\pm 0.86}$	$62.60_{\pm 0.79}$	$68.74_{\pm 0.91}$	**4.49**$_{\pm 0.60}$	$73.59_{\pm 1.45}$	$83.94_{\pm 0.33}$	$8.75_{\pm 2.16}$
CODA-Prompt	$71.85_{\pm 0.54}$	$75.90_{\pm 0.32}$	**3.97**$_{\pm 0.30}$	$66.58_{\pm 0.51}$	$71.99_{\pm 0.54}$	$\underline{4.61}_{\pm 0.44}$	$77.90_{\pm 0.96}$	$84.20_{\pm 0.77}$	$\underline{4.23}_{\pm 0.58}$
EASE	$75.86_{\pm 0.32}$	$82.00_{\pm 0.26}$	$8.20_{\pm 0.40}$	$74.03_{\pm 0.57}$	$80.65_{\pm 0.29}$	$8.04_{\pm 0.27}$	$83.01_{\pm 0.59}$	$89.27_{\pm 1.20}$	$10.27_{\pm 1.09}$
SLCA	$81.14_{\pm 0.02}$	$85.37_{\pm 0.28}$	$7.16_{\pm 0.25}$	$79.04_{\pm 0.06}$	$84.10_{\pm 0.22}$	$8.98_{\pm 0.30}$	$86.58_{\pm 0.32}$	$91.22_{\pm 0.60}$	$9.37_{\pm 0.17}$
MACIL	$\underline{82.04}_{\pm 0.09}$	$\underline{86.40}_{\pm 0.33}$	$5.74_{\pm 0.46}$	$\underline{79.53}_{\pm 0.13}$	$\underline{84.80}_{\pm 0.19}$	$7.11_{\pm 0.30}$	$\underline{90.27}_{\pm 0.40}$	$\underline{93.53}_{\pm 0.27}$	$4.47_{\pm 0.66}$
SLDC	$77.87_{\pm 0.13}$	$84.14_{\pm 0.39}$	$8.48_{\pm 0.40}$	$69.54_{\pm 0.42}$	$80.58_{\pm 0.67}$	$17.43_{\pm 0.61}$	$85.28_{\pm 0.60}$	$91.33_{\pm 0.51}$	$10.04_{\pm 0.52}$
CAMP (ours)	**82.50**$_{\pm 0.27}$	**86.72**$_{\pm 0.14}$	$\underline{4.08}_{\pm 0.23}$	**80.67**$_{\pm 0.05}$	**86.51**$_{\pm 0.29}$	$5.22_{\pm 0.34}$	**90.71**$_{\pm 0.06}$	**93.82**$_{\pm 0.17}$	**3.73**$_{\pm 0.41}$

Table 3: Comparison with EFCIL methods on ImageNet-A and CIFAR-100. Best and second-best results are in **bold** and underline, respectively. All reported values are averaged over three random seeds and ± denotes the standard deviation. Per-task accuracy curves are provided in Appendix F.

Method	IN-A (T=10)			IN-A (T=20)			CIFAR-100 (T=10)		
	Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓
L2P	$43.36_{\pm 0.75}$	$52.00_{\pm 0.33}$	$\underline{6.20}_{\pm 0.82}$	$36.58_{\pm 0.46}$	$45.85_{\pm 1.01}$	$\underline{8.02}_{\pm 1.70}$	$83.45_{\pm 1.41}$	$88.55_{\pm 0.46}$	$9.16_{\pm 2.38}$
DualPrompt	$40.51_{\pm 1.25}$	$52.29_{\pm 1.03}$	$8.45_{\pm 1.21}$	$36.36_{\pm 2.07}$	$48.22_{\pm 1.11}$	$11.10_{\pm 0.68}$	$84.51_{\pm 0.34}$	$89.27_{\pm 0.68}$	$4.72_{\pm 0.61}$
CODA-Prompt	$47.84_{\pm 0.98}$	$59.55_{\pm 1.14}$	**3.77**$_{\pm 0.89}$	$41.08_{\pm 0.47}$	$54.00_{\pm 1.11}$	**5.79**$_{\pm 1.20}$	$86.51_{\pm 0.13}$	$91.16_{\pm 0.05}$	$4.31_{\pm 0.27}$
EASE	$56.68_{\pm 0.60}$	$67.13_{\pm 0.73}$	$12.50_{\pm 0.72}$	$50.76_{\pm 2.12}$	$62.51_{\pm 2.08}$	$13.69_{\pm 1.66}$	$87.65_{\pm 0.31}$	$92.45_{\pm 0.34}$	$6.50_{\pm 0.57}$
SLCA	$60.24_{\pm 0.93}$	$68.17_{\pm 1.88}$	$14.83_{\pm 1.05}$	$56.97_{\pm 0.98}$	$66.30_{\pm 1.44}$	$18.33_{\pm 0.90}$	$89.83_{\pm 0.30}$	$93.70_{\pm 0.18}$	$7.60_{\pm 0.23}$
MACIL	$\underline{63.31}_{\pm 0.31}$	$\underline{71.58}_{\pm 0.95}$	$10.58_{\pm 0.55}$	$\underline{59.23}_{\pm 0.57}$	$\underline{68.59}_{\pm 0.86}$	$14.48_{\pm 0.58}$	$\underline{91.74}_{\pm 0.14}$	$\underline{94.76}_{\pm 0.16}$	$\underline{3.07}_{\pm 0.25}$
SLDC	$53.37_{\pm 0.87}$	$65.92_{\pm 0.78}$	$16.57_{\pm 0.96}$	$42.95_{\pm 0.82}$	$59.08_{\pm 1.22}$	$23.32_{\pm 0.53}$	$89.50_{\pm 0.50}$	$94.13_{\pm 0.11}$	$6.93_{\pm 0.64}$
CAMP (ours)	**65.02**$_{\pm 0.19}$	**72.91**$_{\pm 0.99}$	$8.46_{\pm 0.49}$	**62.74**$_{\pm 0.51}$	**70.69**$_{\pm 0.91}$	$11.15_{\pm 0.72}$	**92.43**$_{\pm 0.15}$	**94.99**$_{\pm 0.13}$	**2.45**$_{\pm 0.15}$

4 Experiments

4.1 Experimental Setup

Datasets. We conduct experiments on four widely used EFCIL benchmarks: CIFAR-100 (Krizhevsky, 2009), ImageNet-R (Hendrycks et al., 2021a), ImageNet-A (Hendrycks et al., 2021b), and CUB-200 (Wah et al., 2011). CIFAR-100 contains 100 natural object classes, ImageNet-R and ImageNet-A each contain 200 classes derived from ImageNet (Russakovsky et al., 2015), and CUB-200 contains 200 classes of fine-grained bird species. For all datasets, we create different task sequences by dividing the classes into T tasks of equal size (e.g., $T = 10$ tasks with 20 classes each for ImageNet-R).

Baselines and evaluation metrics. We compare CAMP against two groups of methods, both targeting the same pre-trained-model setting. The first comprises established parameter-efficient baselines, namely the prompt-based L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a) and CODA-Prompt (Smith et al., 2023) together with the adapter-based EASE (Zhou et al., 2024). The second comprises the strongest recent state-of-the-art approaches, SLCA (Zhang et al., 2023), MACIL (Wu et al., 2025) and SLDC (Rao et al., 2026). Following the standard evaluation protocol (Rao et al., 2026; Wu et al., 2025), we report the average accuracy $\bar{A} = \frac{1}{T}\sum_{t=1}^{T} A_t$ and the final accuracy Acc. To assess retention, we report forgetting Forg $= \frac{1}{T-1}\sum_{i=1}^{T-1}\big(\max_t A_{i,t} - A_{i,T}\big)$, where $A_{i,t}$ is the accuracy on task i after learning task t. This quantifies the average accuracy drop of each task from its peak (Chaudhry et al., 2018).

Implementation details. We adopt ViT-B/16 (Dosovitskiy, 2020) pre-trained on ImageNet-21K (Deng et al., 2009) as the backbone across all experiments. LoRA (Hu et al., 2022) is applied to the key (W_k) and value (W_v) projection matrices of each self-attention layer with rank $r = 32$. The implementations of existing methods are based on the LAMDA-PILOT (Sun et al., 2025) or their official code. Training uses SGD with momentum 0.9, initial learning rate 0.01, cosine annealing, and batch size 48. We use $\gamma = 0.4$ in FMC and $\beta = 0.3$ in PACR. Appendix F.2 gives a per-method accounting of parameters, training time and storage, and shows that CAMP imposes no additional inference-time storage.

Table 4: Component ablation on three benchmarks. ✓ indicates the component is enabled. All reported values are averaged over three random seeds and ± denotes the standard deviation. Best results in **bold**.

#	Components			IN-R (T=10)			IN-A (T=10)			CUB(T=10)		
	FMC	KSC	PACR	Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓
1				$79.55_{\pm 0.10}$	$85.40_{\pm 0.18}$	$12.90_{\pm 0.22}$	$62.04_{\pm 0.08}$	$71.55_{\pm 1.07}$	$15.69_{\pm 0.74}$	$88.15_{\pm 0.19}$	$92.76_{\pm 0.29}$	$7.91_{\pm 0.50}$
2	✓			$82.08_{\pm 0.11}$	$86.79_{\pm 0.26}$	$6.09_{\pm 0.29}$	$63.77_{\pm 0.11}$	$72.35_{\pm 0.93}$	$11.84_{\pm 0.48}$	$90.26_{\pm 0.20}$	$93.58_{\pm 0.11}$	$4.48_{\pm 0.66}$
3		✓		$79.72_{\pm 0.08}$	$85.56_{\pm 0.18}$	$12.30_{\pm 0.38}$	$63.11_{\pm 0.14}$	$72.12_{\pm 1.21}$	$13.37_{\pm 0.53}$	$89.38_{\pm 0.50}$	$93.29_{\pm 0.27}$	$6.06_{\pm 0.82}$
4			✓	$79.57_{\pm 0.20}$	$85.66_{\pm 0.17}$	$11.39_{\pm 0.40}$	$62.70_{\pm 0.28}$	$72.18_{\pm 0.98}$	$13.71_{\pm 0.32}$	$88.66_{\pm 0.16}$	$92.94_{\pm 0.38}$	$6.99_{\pm 0.44}$
5	✓	✓		$82.11_{\pm 0.18}$	$86.84_{\pm 0.30}$	$6.06_{\pm 0.09}$	$64.19_{\pm 0.51}$	$72.45_{\pm 1.05}$	$11.17_{\pm 0.43}$	$90.26_{\pm 0.18}$	$93.59_{\pm 0.10}$	$4.38_{\pm 0.64}$
6	✓		✓	$82.21_{\pm 0.04}$	$86.73_{\pm 0.23}$	$4.67_{\pm 0.40}$	$64.76_{\pm 0.22}$	$72.86_{\pm 0.91}$	$9.40_{\pm 0.49}$	$90.47_{\pm 0.20}$	$93.78_{\pm 0.17}$	$4.08_{\pm 0.39}$
7		✓	✓	$79.28_{\pm 0.12}$	$85.60_{\pm 0.15}$	$11.24_{\pm 0.28}$	$63.81_{\pm 0.08}$	$72.67_{\pm 1.15}$	$11.14_{\pm 0.77}$	$89.88_{\pm 0.24}$	$93.42_{\pm 0.27}$	$5.02_{\pm 0.34}$
8	✓	✓	✓	**82.50**$_{\pm 0.27}$	**86.72**$_{\pm 0.14}$	**4.08**$_{\pm 0.23}$	**65.02**$_{\pm 0.19}$	**72.91**$_{\pm 0.99}$	**8.46**$_{\pm 0.49}$	**90.71**$_{\pm 0.06}$	**93.82**$_{\pm 0.17}$	**3.73**$_{\pm 0.41}$

4.2 Main Results

Tables 2 and 3 report the comparison across six dataset configurations. Per-task curves are in Appendix F (Figures 3–4). CAMP achieves the best last and average accuracy in every setting. On short sequences ($T = 10$) it surpasses the second-best method MACIL by $+0.46$ on ImageNet-R, $+1.71$ on ImageNet-A and $+0.44$ on CUB-200, and the margin grows on long sequences ($T = 20$): $+1.14$ on ImageNet-R and $+3.51$ on ImageNet-A, the latter a relative gain of roughly 6% over the strongest competitor on this artistic and adversarial benchmark.

CAMP also attains the lowest forgetting on CUB-200 and CIFAR-100. Where another method reports a numerically lower value it does so at substantially lower last accuracy: on ImageNet-A ($T = 20$) CODA-Prompt reaches forgetting 5.79 but only 41.08 last accuracy against CAMP's 62.74. A method that fails to acquire knowledge has nothing to forget, so its low forgetting reflects poor plasticity rather than strong retention. Among methods of comparable accuracy, CAMP reduces forgetting by a wide margin in every setting.

4.3 Ablation Studies

4.3.1 Effect of different components.

Table 4 reports the contribution of each component across all eight combinations. Relative to the baseline (#1), every individual component improves performance, but their magnitudes differ sharply and they act on different metrics. FMC alone (#2) provides the largest single-component gain: on ImageNet-R it raises last accuracy by 2.53 and lowers forgetting by 6.81, far exceeding KSC alone (#3, $+0.17$ Acc, -0.60 Forg) and PACR alone (#4, $+0.02$ Acc, -1.51 Forg). FMC is therefore the structural prerequisite of the pipeline: without a stabilized manifold, neither prototype correction nor metric recalibration can operate effectively. On retention, KSC and PACR contribute the bulk of the remaining improvement: relative to the baseline, Forg falls by 68%, 46% and 53% across the three benchmarks, and PACR alone adds a further -1.98, -2.71 and -0.65 on top of FMC+KSC (#5 $\rightarrow$ #8). From FMC alone to the full pipeline, last accuracy improves by only 0.42, 1.25 and 0.45 points, while Forg falls by 2.01, 3.38 and 0.75. The accuracy deltas are small in absolute terms, but they are consistent across all three random seeds and are therefore a genuine effect rather than noise, while the retention gains are an order of magnitude larger. This asymmetry is precisely what the causal pipeline predicts: FMC stabilizes the manifold and determines how much can be learned, whereas KSC and PACR operate on the already-learned representation and determine how much of it is retained.

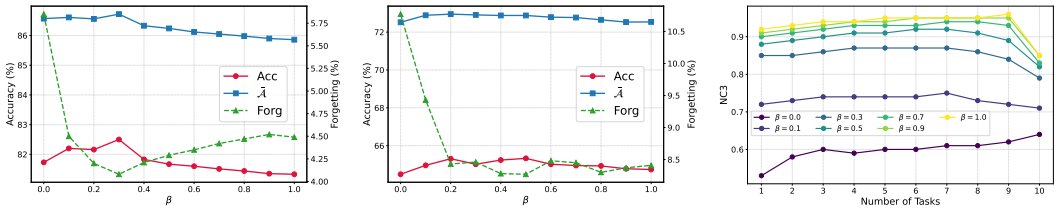
Consistent with this reading, KSC and PACR without FMC (#7) yield no gain over either alone: on ImageNet-R, KSC+PACR actually degrades accuracy relative to KSC alone (79.28 vs. 79.72), because both require the stable manifold that FMC provides. Conversely, adding KSC or PACR on top of FMC (#5 and #6) yields only marginal accuracy gains over FMC alone, with their benefit appearing almost entirely in forgetting. Adding either on top of FMC (#5, #6) improves retention further, and the full pipeline (#8) attains the best accuracy and the lowest forgetting on every benchmark. The three components are thus not additive but interdependent stages of a causal pipeline: FMC constrains the manifold, KSC corrects residual coordinate drift on that stable foundation, and PACR recalibrates the metric to lock in the gains.

4.3.2 Different strategies of distillation

Table 5 ablates the choice of distillation objectives within FMC on ImageNet-R and ImageNet-A. As shown in Table 5, both CLS alone (#2) and Patch alone (#3) improve substantially over the base-

Table 5: FMC distillation ablation. All variants include KSC and PACR. Default setting of FMC shown in **bold**. All reported values are averaged over three random seeds and ± denotes the standard deviation.

#	Strategy	ImageNet-R			ImageNet-A		
		Acc↑	$\bar{A}$↑	Forg↓	Acc↑	$\bar{A}$↑	Forg↓
1	Baseline	$79.28_{\pm 0.12}$	$85.60_{\pm 0.15}$	$11.24_{\pm 0.28}$	$63.81_{\pm 0.08}$	$72.67_{\pm 1.15}$	$11.14_{\pm 0.77}$
2	CLS	$82.18_{\pm 0.13}$	$86.81_{\pm 0.23}$	$4.81_{\pm 0.21}$	$64.91_{\pm 0.88}$	$72.82_{\pm 1.30}$	$8.85_{\pm 0.49}$
3	Patch	$81.46_{\pm 0.38}$	$86.77_{\pm 0.32}$	$6.99_{\pm 0.19}$	$64.33_{\pm 0.49}$	$72.09_{\pm 1.14}$	$9.32_{\pm 0.30}$
4	**CLS + Patch**	$\mathbf{82.50}_{\pm 0.27}$	$\mathbf{86.72}_{\pm 0.14}$	$\mathbf{4.08}_{\pm 0.23}$	$\mathbf{65.02}_{\pm 0.19}$	$\mathbf{72.91}_{\pm 0.99}$	$\mathbf{8.46}_{\pm 0.49}$
5	CLS + Patch + Norm	$82.54_{\pm 0.30}$	$86.55_{\pm 0.13}$	$4.09_{\pm 0.19}$	$64.93_{\pm 0.20}$	$72.90_{\pm 0.98}$	$8.58_{\pm 0.49}$
6	CLS + Patch + Sim	$82.47_{\pm 0.26}$	$86.52_{\pm 0.14}$	$4.01_{\pm 0.26}$	$64.73_{\pm 0.27}$	$72.85_{\pm 0.97}$	$8.58_{\pm 0.71}$



(a) ImageNet-R (10 steps) (b) ImageNet-A (10 steps) (c) ImageNet-R per-task NC3

Figure 2: (a)–(b) Sensitivity of Acc, $\bar{A}$, and Forg to β on ImageNet-R and ImageNet-A. (c) Per-task NC3 on ImageNet-R for selected β values, with lighter colors indicating larger β.

line (#1) on both datasets, confirming that each individual distillation target provides meaningful manifold regularization. CLS + Patch (#4) outperforms either alone, with particularly pronounced gains on ImageNet-R (+2.90 Acc for CLS alone, +2.18 for Patch alone, and +3.22 for their combination). This demonstrates that the two objectives capture complementary gradient subspaces and that neither alone is sufficient for full manifold preservation.

In contrast, adding Norm (#5) or Sim (#6) on top of CLS + Patch yields virtually no additional gain. On ImageNet-R, CLS + Patch + Norm differs from CLS + Patch by only +0.04 Acc, and CLS + Patch + Sim by −0.03 Acc, both well within the standard deviation of the measurement. The same pattern holds on ImageNet-A. These results directly validate the gradient subspace analysis of Section 3.2: CLS + Patch already spans the full gradient space of ViT feature-level distillation, and Norm and Sim are strict gradient subsets that confer zero independent benefit.

4.3.3 Sensitivity of β

Figure 2 examines the sensitivity of CAMP to the re-anchoring strength β. Performance is stable over $\beta \in [0.2, 0.4]$ on both datasets, with $\beta = 0.3$ giving the best overall trade-off: disabling PACR ($\beta = 0$) makes forgetting rise sharply, while $\beta = 1.0$ discards the discriminative structure encoded in w_c^{t-1}, an effect more pronounced on ImageNet-R where inter-class geometry matters more. Figure 2 (c) shows the same trade-off through Neural Collapse (Papyan et al., 2020): NC3, the alignment $\frac{1}{C}\sum_c \cos(w_c, \mu_c)$ between classifier weights and prototypes, stays low without PACR and rises with β, yet the highest NC3 ($\beta = 1.0$) does not give the best accuracy, because pure prototype alignment discards the relative positioning encoded in w_c and over-tightens the classifier around the current prototypes, which effectively curbs the plasticity needed to accommodate new classes.

5 Conclusion

We presented CAMP, which treats manifold preservation and classifier recalibration as two sides of the same geometric problem and interlinks them as a causal pipeline rather than independent stages. CAMP combines FMC (gradient-complete manifold constraint), KSC (non-parametric minimax-optimal drift correction) and PACR (pullback-metric re-anchoring). Experiments show state-of-the-art accuracy with substantially improved retention, confirming that coupling these problems yields gains beyond treating them independently and pointing toward a principled geometric approach to EFCIL.

AI USE STATEMENT

In this work, we used generative AI tools for one task with recommended disclosure: editing this research paper to improve its readability, i.e., grammar, clarity, and language polishing of text that we had already written. We have not used generative AI tools for any of the tasks with required disclosure, and the remaining required-disclosure tasks are not applicable to this work. We take full responsibility for the final content of this work, including all text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

We have made extensive efforts to ensure that our results can be reproduced. The complete source code is provided as an anonymous ZIP archive in the supplementary materials. A detailed description of the datasets, network architectures, and hyperparameters is given in Section 4, with the full experimental settings in Appendix F.

REFERENCES

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 532–547, 2018.

Faris Chaudhry. The geometry of projection heads: Conditioning, invariance, and collapse. *arXiv preprint arXiv:2605.17180*, 2026.

Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *International Conference on Machine Learning*, pp. 1952–1961. PMLR, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Vassiliy A Epanechnikov. Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158, 1969.

Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. 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.

Alex Gomez-Villa, Dipam Goswami, Kai Wang, Andrew D Bagdanov, Bartlomiej Twardowski, and Joost Van De Weijer. Exemplar-free continual representation learning via learnable drift compensation. In *European Conference on Computer Vision*, pp. 473–490. Springer, 2024.

Jiangpeng He, Zhihao Duan, and Fengqing Zhu. Cl-lora: Continual low-rank adaptation for rehearsal-free class-incremental learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 30534–30544, 2025a.

Run He, Di Fang, Yicheng Xu, Yawen Cui, Ming Li, Cen Chen, Ziqian Zeng, and Huiping Zhuang. Semantic shift estimation via dual-projection and classifier reconstruction for exemplar-free class-incremental learning. *arXiv preprint arXiv:2503.05423*, 2025b.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15262–15271, 2021b.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in neural information processing systems*, 32, 2019.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

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*, pp. 28495–28504, 2024.

Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International conference on machine learning*, pp. 3925–3934. PMLR, 2019.

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

Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23638–23647, 2024.

Xuandi Luo, Huaidong Zhang, Yi Xie, and Shengfeng He. Saber: Continual learning with representation conflict management. In *Forty-third International Conference on Machine Learning*, 2026.

Simone Magistri, Tomaso Trinci, Joost van de Weijer, Andrew Bagdanov, et al. Elastic feature consolidation for cold start exemplar-free incremental learning. In *International Conference on Learning Representations*, volume 2024, pp. 31909–31931, 2024.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40): 24652–24663, 2020.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

Xuan Rao, Simian Xu, Zheng Li, Bo Zhao, Derong Liu, Mingming Ha, and Cesare Alippi. Compensating distribution drifts in continual learning with pre-trained vision transformers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(30):25090–25098, Mar. 2026. doi: 10.1609/aaai.v40i30.39698. URL `https://ojs.aaai.org/index.php/AAAI/article/view/39698`.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Grzegorz Rypeść, Sebastian Cygert, Tomasz Trzciński, and Bartłomiej Twardowski. Task-recency bias strikes back: Adapting covariances in exemplar-free class incremental learning. In *Advances in Neural Information Processing Systems*, 2024.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11909–11919, 2023.

Hai-Long Sun, Da-Wei Zhou, De-Chuan Zhan, and Han-Jia Ye. Pilot: A pre-trained model-based continual learning toolbox, 2025.

Yuwen Tan, Qinhao Zhou, Xiang Xiang, Ke Wang, Yuchuan Wu, and Yongbin Li. Semantically-shifted incremental adapter-tuning is a continual vitransformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23252–23262, 2024.

Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1365–1374, 2019.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Matt P Wand and M Chris Jones. *Kernel Smoothing*. Chapman and Hall/CRC, 1995.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. *arXiv preprint arXiv:2310.14152*, 2023.

Xiequn Wang, Zhan Zhuang, and Yu Zhang. Plan: Proactive low-rank allocation for continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2909–2918, 2025.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision*, pp. 631–648. Springer, 2022a.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 139–149, 2022b.

Fangwen Wu, Lechao Cheng, Shengeng Tang, Xiaofeng Zhu, Chaowei Fang, Dingwen Zhang, and Meng Wang. Navigating semantic drift in task-agnostic class-incremental learning. In *Forty-second International Conference on Machine Learning*, 2025.

Zhendong Yang, Zhe Li, Ailing Zeng, Zexian Li, Chun Yuan, and Yu Li. Vitkd: Feature-based knowledge distillation for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1379–1388, 2024.

Lu Yu, Bartlomiej 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*, pp. 6982–6991, 2020.

Jia-Tong Zhai, Xialei Liu, Lu Yu, and Ming-Ming Cheng. Fine-grained knowledge selection and restoration for non-exemplar class incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 6971–6978, 2024.

Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. 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.

Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. 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.

Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 133(3):1012–1032, 2025.

A ALGORITHM PIPELINE

Algorithm 1 CAMP Training Pipeline

Input: Training dataset $\{\mathcal{D}_t\}_{t=0}^{T-1}$, pre-trained backbone f_{θ_0}, hyperparameters γ, β
Output: Trained backbone f_{θ_T}, classifier $W^{(T)}$

```
Initialize classifier W^(t)
for t = 1 to T do
    // Training backbone
    Expand classifier for C_t, initialize weights
    L_total = L_CE if t = 1, else L_total = L_CE + L_FMC with Eq. equation 9
    for each batch (x, y) ∈ D_t do
        Compute L_total (using f_{θ_{t-1}} if t > 1), update θ_t
    end for
    Compute and store {μ_c^(t), Σ_c^(t)}_{c∈C_t} via Eq. equation 10 and equation 11
    if t > 1 then
        Compute drifts {Δf(x_i)}_{x_i∈D_t} with Eq. equation 12
        Update old prototypes {μ̂_c^(t)}_{c∈C_{1:t-1}} via KSC with Eq. equation 13 and equation 14
        Re-anchor classifier weight {w_c}_{c∈C_{1:t-1}} via PACR with Eq. equation 15
        // Fine-tune classifier
        for each c ∈ C_{1:t-1} do
            Sample {x̃} ~ N(μ̂_c^(t), Σ_c^(t-1))
            Train classifier W^(t) on {x̃} with L_CE
        end for
    end if
end for
```

B FMC GRADIENT SUBSPACE ANALYSIS

In this appendix, we provide the full gradient derivations for the four distillation losses discussed in Section 3.2 of the main paper, and establish the subspace inclusion relations that prove CLS + Patch forms the minimal sufficient distillation basis for manifold preservation.

Let the ViT backbone f_θ produce CLS token $c = \sum_{i=1}^{N} w_i p_i \in \mathbb{R}^d$ and patch tokens $\{p_i\}_{i=1}^{N} \subset \mathbb{R}^d$, where w_i are the attention weights from the final self-attention layer aggregating patch tokens into the CLS token. The backbone parameters are denoted by θ. For an input x, the old backbone $f_{\theta_{t-1}}$ produces $(c^{(t-1)}, \{p_i^{t-1}\})$ and the new backbone f_{θ_t} produces $(c^{(t)}, \{p_i^t\})$. We omit the task subscripts and analyze four distillation losses:

$$\mathcal{L}_{\text{CLS}} = \|c - c^{(t-1)}\|^2, \tag{17}$$

$$\mathcal{L}_{\text{Patch}} = \sum_{i=1}^{N} \|p_i - p_i^{t-1}\|^2, \tag{18}$$

$$\mathcal{L}_{\text{Norm}} = (\|c\| - \|c^{(t-1)}\|)^2, \tag{19}$$

$$\mathcal{L}_{\text{Sim}} = \|M - M^{(t-1)}\|_F^2, \quad M_{ij} = \frac{p_i \cdot p_j}{\|p_i\|\|p_j\|}. \tag{20}$$

B.1 Gradient of CLS Loss

The CLS loss gradient with respect to backbone parameters θ expands via $c = \sum_i w_i p_i$:

$$\begin{aligned}\frac{\partial \mathcal{L}_{\text{CLS}}}{\partial \theta} &= 2(c - c^{(t-1)})^\top \frac{\partial c}{\partial \theta} \\ &= 2(c - c^{(t-1)})^\top \Big(\sum_i w_i \frac{\partial p_i}{\partial \theta} + \sum_i p_i \frac{\partial w_i}{\partial \theta}\Big) \\ &= \underbrace{2(c - c^{(t-1)})^\top \sum_i w_i \frac{\partial p_i}{\partial \theta}}_{\text{(a) Weighted per-token position gradient}} \\ &\quad + \underbrace{2(c - c^{(t-1)})^\top \sum_i p_i \frac{\partial w_i}{\partial \theta}}_{\text{(b) Attention weight gradient}}.\end{aligned} \tag{21}$$

Term (a) provides gradient signals to each patch token p_i, weighted by its attention coefficient w_i. Term (b) is unique to $\mathcal{L}_{\text{CLS}}$, governing how the attention aggregation mechanism itself changes with θ. No other distillation loss constrains $\partial w_i / \partial \theta$.

B.2 Gradient of Patch Loss

$$\frac{\partial \mathcal{L}_{\text{Patch}}}{\partial \theta} = \sum_i 2(p_i - p_i^{t-1})^\top \frac{\partial p_i}{\partial \theta}. \tag{22}$$

This gradient decomposes into three sub-components:

- **Per-token position**: the full vector $2(p_i - p_i^{t-1})$ directly constrains the position of each patch token in $\mathbb{R}^d$.
- **Per-token magnitude**: the L_2 distance $\|p_i - p_i^{t-1}\|^2$ decomposes into $\|p_i\|^2 + \|p_i^{t-1}\|^2 - 2\|p_i\|\|p_i^{t-1}\| \cos\alpha$, so minimizing it forces $\|p_i\| \to \|p_i^{t-1}\|$, implicitly preserving magnitudes.
- **Per-token direction**: similarly, the L_2 distance is minimized only when $\cos\alpha \to 1$, so the unit direction $p_i/\|p_i\|$ is also implicitly constrained.

B.3 Gradient of Norm Loss

$$\frac{\partial \mathcal{L}_{\text{Norm}}}{\partial \theta} = 2(\|c\| - \|c^{(t-1)}\|) \cdot \frac{c^\top}{\|c\|} \frac{\partial c}{\partial \theta}. \tag{23}$$

Substituting the expansion of $\partial c/\partial\theta$ from Eq. equation 21:

$$\frac{\partial \mathcal{L}_{\text{Norm}}}{\partial \theta} = \frac{2(\|c\| - \|c^{(t-1)}\|)}{\|c\|} \cdot c^\top \sum_i \Big(w_i \frac{\partial p_i}{\partial \theta} + p_i \frac{\partial w_i}{\partial \theta}\Big). \tag{24}$$

Comparing with Eq. equation 21, both parameter gradients share the same Jacobian $J = \partial c/\partial\theta$ and differ only in the feature-space direction that weights it: $\mathcal{L}_{\text{Norm}}$ weights it by $2(\|c\| - \|c^{(t-1)}\|)\, c/\|c\|$, which is always parallel to c, whereas $\mathcal{L}_{\text{CLS}}$ weights it by $2(c - c^{(t-1)})$. As the target $c^{(t-1)}$ varies over the data, $2(c - c^{(t-1)})$ sweeps all of $\mathbb{R}^d$, so every feature-space direction reachable by $\mathcal{L}_{\text{Norm}}$ is already reachable by $\mathcal{L}_{\text{CLS}}$:

$$\Big\{\frac{\partial \mathcal{L}_{\text{Norm}}}{\partial c}\Big\} \subseteq \text{span}\Big\{\frac{\partial \mathcal{L}_{\text{CLS}}}{\partial c}\Big\} \qquad \text{(feature space)}. \tag{25}$$

We emphasize that this inclusion is stated on $\partial\mathcal{L}/\partial c$ and not on $\partial\mathcal{L}/\partial\theta$. Because the shared Jacobian J is applied to generally non-collinear weight vectors, $\partial\mathcal{L}_{\text{Norm}}/\partial\theta$ does not in general lie in the span of a single instantiation of $\partial\mathcal{L}_{\text{CLS}}/\partial\theta$. The question relevant for redundancy, however, is whether $\mathcal{L}_{\text{Norm}}$ can constrain a feature direction that $\mathcal{L}_{\text{CLS}}$ leaves unconstrained, and the answer is negative.

B.4 Gradient of Similarity Loss

The gradient through each patch token p_i expands as:

$$\begin{aligned}\frac{\partial \mathcal{L}_{\text{Sim}}}{\partial \theta} &= \sum_i \frac{\partial \mathcal{L}_{\text{Sim}}}{\partial p_i}\frac{\partial p_i}{\partial \theta} \\ &= \sum_i \Big(2\sum_{j\neq i}(M_{ij} - M_{ij}^{(t-1)}) \cdot \frac{\partial M_{ij}}{\partial p_i}\Big)\frac{\partial p_i}{\partial \theta},\end{aligned} \tag{26}$$

where

$$\begin{aligned}\frac{\partial M_{ij}}{\partial p_i} &= \frac{p_j}{\|p_i\|\|p_j\|} - \frac{(p_i \cdot p_j)\, p_i}{\|p_i\|^3\|p_j\|} \\ &= \frac{1}{\|p_i\|}\Big(\frac{p_j}{\|p_j\|} - M_{ij}\frac{p_i}{\|p_i\|}\Big).\end{aligned} \tag{27}$$

The gradient of $\mathcal{L}_{\text{Sim}}$ depends only on the unit-normalized directions $p_i/\|p_i\|$ and $p_j/\|p_j\|$. The magnitude $\|p_i\|$ appears only in the scalar factor $1/\|p_i\|$ and does not contribute an independent gradient component. Consequently, $\mathcal{L}_{\text{Sim}}$ provides no independent constraint on per-token magnitudes beyond what is already encoded in direction matching. Thus, we have

$$\frac{\partial \mathcal{L}_{\text{Patch}}}{\partial \theta} \supset \frac{\partial \mathcal{L}_{\text{Sim}}}{\partial \theta}. \tag{28}$$

Since $\partial \mathcal{L}_{\text{Sim}}/\partial p_i$ depends only on patch directions while $\partial \mathcal{L}_{\text{Patch}}/\partial p_i$ involves the full vector, the same inclusion propagates to θ via the chain rule.

B.5 Subspace Inclusion Hierarchy

Combining the above analyses, the gradient subspace inclusion hierarchy is:

$$\begin{gathered}\underbrace{\nabla_\theta \mathcal{L}_{\text{CLS}}}_{(a)+(b)} \supset \nabla_\theta \mathcal{L}_{\text{Norm}} \\ \underbrace{\nabla_\theta \mathcal{L}_{\text{Patch}}}_{(c)} \supset \nabla_\theta \mathcal{L}_{\text{Sim}} \\ \nabla_\theta \mathcal{L}_{\text{CLS}} \cap \nabla_\theta \mathcal{L}_{\text{Patch}} = \{(a)\} \quad \text{(shared: position)} \\ \nabla_\theta \mathcal{L}_{\text{CLS}} \setminus \nabla_\theta \mathcal{L}_{\text{Patch}} = \{(b)\} \quad \text{(unique: attention)} \\ \nabla_\theta \mathcal{L}_{\text{Patch}} \setminus \nabla_\theta \mathcal{L}_{\text{CLS}} = \{\text{magnitude, direction}\}\end{gathered}$$

Therefore, $\nabla_\theta \mathcal{L}_{\text{CLS}} \cup \nabla_\theta \mathcal{L}_{\text{Patch}}$ spans the full distillation gradient subspace identified in Table 1 of the main paper, while $\nabla_\theta \mathcal{L}_{\text{CLS}} \cup \nabla_\theta \mathcal{L}_{\text{Sim}}$ omits per-token magnitude and $\nabla_\theta \mathcal{L}_{\text{Patch}} \cup \nabla_\theta \mathcal{L}_{\text{Norm}}$ omits attention aggregation gradients. $\mathcal{L}_{\text{CLS}} + \mathcal{L}_{\text{Patch}}$ is the minimal sufficient distillation set.

C From Patch-Token Structure to Prototype Stability

Section 3.2 shows that CLS and patch distillation jointly form the minimal sufficient constraint on the feature manifold. Since KSC and PACR operate on class-level prototypes rather than on individual tokens, we make explicit here how a patch-level constraint propagates to prototype stability.

Write the per-sample feature as the CLS token $f_\theta(x) = c(x) = \sum_j w_j(x)\, p_j(x)$ (Eq. 5), and let $\mu_c = \frac{1}{N_c}\sum_{i:y_i=c} c(x_i)$ be the prototype of class c. The drift of the prototype between consecutive tasks decomposes into two contributions,

$$\|\mu_c^{(t)} - \mu_c^{(t-1)}\| \;\le\; \underbrace{\big\|\bar{c}^{(t)} - \bar{c}^{(t-1)}\big\|}_{\text{CLS drift}} + \underbrace{\big\|\bar{c}_{\text{patch}}^{(t)} - \bar{c}_{\text{patch}}^{(t-1)}\big\|}_{\text{within-class patch coherence}}, \tag{29}$$

where $\bar{\cdot}$ denotes the class mean and $\bar{c}_{\text{patch}}$ the attention-weighted patch component. The CLS term of FMC controls the first contribution directly. The second is not controlled by the CLS term alone: matching class means does not constrain whether the within-class distribution of patch tokens is preserved, so patch tokens may drift incoherently across samples of the same class while each individual CLS token remains well matched.

This is exactly what the patch term prevents. $\mathcal{L}_{\text{Patch}}$ penalizes $\|p_j(x) - p_j^{(t-1)}(x)\|$ for every sample and every patch index j, so the drift of a given patch index is constrained to be consistent across

samples of a class: all samples shift their j-th patch in a similar direction. Averaging over the class therefore does not cancel the correction, and the prototype moves coherently instead of accumulating an incoherent, variance-inflated residue.

The consequence for the later stages is direct. KSC estimates the prototype shift as a kernel-weighted average of per-sample embedding shifts $\Delta f(x_i)$ (Eq. equation 13). Incoherent within-class shifts inflate the variance of that weighted average, whereas the coherence enforced by the patch term keeps the estimator $\hat{\mu}_c^{(t)}$ well behaved. PACR then anchors the classifier to $\hat{\mu}_c^{(t)}$ (Eq. equation 15), so a lower-variance prototype estimate tightens the alignment bound of Appendix E. In short, the patch term governs the within-class coherence of the drift field, which is precisely the quantity the class-level stages consume.

D Epanechnikov Kernel Minimax Optimality

In this appendix, we provide the optimality analysis of the kernel selection in KSC. KSC estimates each old prototype's drift via the Nadaraya-Watson (NW) estimator (Wand & Jones, 1995), a classic nonparametric regression method that computes a locally weighted average: samples closer to the query point receive higher weight via a kernel function K, and the estimated value is the kernel-weighted average of the observed responses.

$$\widehat{\Delta\mu}_c = \frac{\sum_i K\Big(\frac{\|f_{\theta_{t-1}}(x_i)-\mu_c\|}{h}\Big)\,\Delta f(x_i)}{\sum_i K\Big(\frac{\|f_{\theta_{t-1}}(x_i)-\mu_c\|}{h}\Big)}, \tag{30}$$

where $h > 0$ is the bandwidth, K is a non-negative kernel with $\int K(u)du = 1$, and $\Delta f(x_i) = f_{\theta_t}(x_i) - f_{\theta_{t-1}}(x_i)$ is the embedding shift of sample x_i.

D.1 Where the Estimation Error Comes From

To analyze which kernel is optimal, we examine the estimation error of the NW estimator. At a query point z, the mean squared error $\mathrm{MSE}(z) = \mathbb{E}[(\widehat{\Delta\mu}_c - \Delta\mu_c^*)^2 \mid z]$ measures the expected squared deviation of the estimate from the true drift. Under standard smoothness conditions, the MSE admits the classical bias-variance decomposition (Wand & Jones, 1995):

$$\begin{aligned}\mathrm{MSE}(z) = &\underbrace{\frac{h^4}{4}\,\sigma_K^4\,B(z)}_{\text{squared bias}} + \underbrace{\frac{R(K)}{N_t h^d}\,V(z)}_{\text{variance}} \\ &+ o\Big(h^4 + \tfrac{1}{N_t h^d}\Big),\end{aligned} \tag{31}$$

where $\sigma_K^2 = \int u^2 K(u)du$, $R(K) = \int K(u)^2 du$, and $B(z), V(z)$ depend only on the data distribution, not on K. The squared bias grows with h^4 (larger bandwidth smooths out local structure), while the variance decays with $N_t h^d$ (more samples reduce noise).

To obtain a global measure of kernel quality, we integrate $\mathrm{MSE}(z)$ over the feature space, yielding the Mean Integrated Squared Error (MISE). Taking the asymptotic limit ($N_t \to \infty$, $h \to 0$, $N_t h^d \to \infty$) and optimizing the bandwidth h gives the Asymptotic MISE (AMISE):

$$\mathrm{AMISE}_{\mathrm{opt}}(K) \propto \big[R(K)^4 \cdot \sigma_K^{2d}\big]^{1/(d+4)}. \tag{32}$$

Thus the kernel influences the optimal error only through $R(K) = \int K(u)^2 du$, the integrated squared roughness of the kernel.

D.2 Why Epanechnikov is Optimal

Since the kernel matters only through $R(K)$, the optimal kernel is the one that minimizes $R(K)$. We seek a non-negative kernel with compact support on $[-1, 1]$, normalized ($\int K = 1$), and continuous at the boundary ($K(1) = 0$). Minimizing $\int K(u)^2 du$ subject to these constraints via the Euler-Lagrange equation gives $K(u) = \lambda_1 + \lambda_2 u^2$ on $[-1, 1]$. The boundary and normalization conditions fix the coefficients, yielding the Epanechnikov kernel:

$$K_{\mathrm{epa}}(u) = \frac{3}{4}(1 - u^2) \cdot \mathbf{1}_{|u|\le 1}, \tag{33}$$

with $R(K_{\text{epa}}) = 3/5$. Among all non-negative kernels, K_{epa} achieves the smallest possible AMISE, i.e., it is minimax optimal over the Hölder class of drift functions (Epanechnikov, 1969).

D.3 Comparison with Other Kernels

To see the practical significance, consider the three kernels commonly used for density estimation:

$$K_{\text{gauss}}(u) = \frac{1}{\sqrt{2\pi}} e^{-u^2/2}, \tag{34}$$

$$K_{\text{lap}}(u) = \frac{1}{2} e^{-|u|}, \tag{35}$$

$$K_{\text{epa}}(u) = \frac{3}{4}(1-u^2) \cdot \mathbf{1}_{|u|\le 1}. \tag{36}$$

To compare kernels fairly, we examine the quantity that appears in the optimized AMISE: the product $R(K) \cdot \sigma_K$ (a smaller value means a smaller variance contribution in equation 31). We compute this quantity for three common kernels.

Epanechnikov. $K(u) = \frac{3}{4}(1-u^2) \cdot \mathbf{1}_{|u|\le 1}$.

$$\sigma_K^2 = \int_{-1}^{1} \tfrac{3}{4} u^2 (1-u^2)\, du = \frac{1}{5}, \quad \sigma_K = \frac{1}{\sqrt{5}}.$$

$$R(K) = \int_{-1}^{1} \tfrac{9}{16} (1-u^2)^2\, du = \frac{3}{5}.$$

$$R \cdot \sigma_K = \frac{3}{5\sqrt{5}} \approx 0.268. \tag{37}$$

Gaussian. $K(u) = \frac{1}{\sqrt{2\pi}} e^{-u^2/2}$, $\sigma_K = 1$.

$$R(K) = \int_{-\infty}^{\infty} \frac{1}{2\pi} e^{-u^2}\, du = \frac{1}{2\sqrt{\pi}}, \tag{38}$$

$$R \cdot \sigma_K = \frac{1}{2\sqrt{\pi}} \approx 0.282. \tag{39}$$

Laplacian. $K(u) = \frac{1}{2} e^{-|u|}$.

$$\sigma_K^2 = \int_{-\infty}^{\infty} \tfrac{1}{2} u^2 e^{-|u|}\, du = 2, \quad \sigma_K = \sqrt{2}.$$

$$R(K) = \int_{-\infty}^{\infty} \tfrac{1}{4} e^{-2|u|}\, du = \frac{1}{4}.$$

$$R \cdot \sigma_K = \frac{\sqrt{2}}{4} \approx 0.354. \tag{40}$$

Epanechnikov achieves the smallest product (0.268), about 5% below Gaussian (0.282) and 24% below Laplacian (0.354), minimizing the variance term and thereby the AMISE. No other non-negative kernel can improve upon this guarantee.

D.4 Compact Support Matters in Practice

A second advantage of Epanechnikov is compact support: $K_{\text{epa}}(u) = 0$ for $|u| > 1$, meaning samples beyond the calibrated bandwidth contribute exactly zero weight. The Gaussian kernel, in contrast, assigns exponentially small but non-zero weight to every sample, regardless of distance. In the EFCIL setting, this distinction matters: drift patterns are class-specific, and distant samples from unrelated classes carry no useful information about the prototype shift of class c. Compact support naturally enforces this locality, while Gaussian tails accumulate noise from irrelevant samples, which can introduce non-trivial bias in high-dimensional feature spaces ($d = 768$).

D.5 Scaling to d Dimensions, Bandwidth Choice, and Assumptions

From one dimension to d. The derivations above are written for a univariate kernel, whereas KSC evaluates K on the scalar distance $\|f_{\theta_{t-1}}(x_i) - \mu_c\|/\sigma$ in a d-dimensional feature space ($d =$

768). This is a radial kernel, and the estimator in Eq. equation 30 is a ratio. Consequently the dimension-dependent normalization constant of a radial kernel, $c_d = 1/\int_{\|u\|\le 1} K(\|u\|)\,du$, occurs as a common factor in both numerator and denominator and cancels exactly: the KSC estimate is invariant to it, and no d-dependent rescaling is needed in the implementation. The dimension does enter the asymptotic analysis, through the moment σ_K^2 and the exponent $1/(d+4)$ of Eq. equation 32, but it affects only the optimal bandwidth and not the ordering of kernels: that comparison is driven by $R(K) = \int K(u)^2 du$, whose ranking (Epanechnikov $<$ Gaussian $<$ Laplacian) is dimension-independent.

Assumptions. The minimax statement requires the standard regularity conditions of non-parametric regression. The drift field is β-Hölder smooth with $\beta \ge 2$, and the design density is bounded away from zero and infinity. The kernel is non-negative, bounded and compactly supported, and the bandwidth satisfies $h \to 0$ with $N_t h^d \to \infty$. Under these conditions the AMISE-optimal non-negative compactly supported kernel is Epanechnikov. These guarantees are asymptotic, and in finite samples the bias term (kernel-independent to leading order but bandwidth-dependent) can dominate, so the practical advantage of Epanechnikov over Gaussian is expected to be modest. This is consistent with our finding that the three kernels perform comparably at the sample sizes used here.

Bandwidth choice. KSC sets σ to the median distance between current-task samples and the old prototype $\mu_c^{(t-1)}$. Because σ is recomputed per class, it behaves as an adaptive radius rather than a fixed one: for a class whose samples lie close to the prototype, σ is small and only a tight neighbourhood contributes, whereas for a widely scattered class σ grows and the effective neighbourhood widens accordingly. In the extreme case where every current sample is far from $\mu_c^{(t-1)}$, roughly half of them remain inside the bandwidth by construction, so compact support alone does not guarantee that the selected samples are semantically related to class c. Two properties bound the resulting error: (i) the weights decrease monotonically with distance, so the closest samples dominate the weighted average wherever the cutoff falls, and (ii) FMC keeps the within-class embedding shifts coherent (Appendix C), so even a widened neighbourhood averages shifts that point in a consistent direction instead of cancelling. KSC is thus not relied upon as a standalone semantic filter, but as one stage of a pipeline whose first stage stabilizes the drift field it operates on.

E Theoretical Justification of PACR

In this appendix, we provide the theoretical foundations for PACR. We first formalize the pullback metric induced by the classifier and characterize how manifold deformation disrupts it. We then derive an alignment lower bound that jointly incorporates manifold preservation (ε) and prototype correction (δ), and show that this single bound explains why FMC is necessary, why FMC+KSC+PACR is synergistic, and why a small β suffices. Finally, we close with a connection to Neural Collapse theory.

E.1 Pullback Metric of Classifier

We first formalize the geometric relationship between classifier weights and the feature manifold. Let $f_\theta : \mathcal{X} \to \mathcal{M} \subset \mathbb{R}^d$ be a backbone mapping inputs to a d-dimensional feature manifold $\mathcal{M}$. A linear classifier $W = [w_1, \dots, w_C] \in \mathbb{R}^{d\times C}$ assigns class scores $s_c(x) = w_c^\top f_\theta(x)$, partitioning $\mathcal{M}$ into C decision regions via the pullback metric

$$g_W(u, v) = u^\top (WW^\top) v, \quad \forall u, v \in T_p\mathcal{M}, \tag{41}$$

where $T_p\mathcal{M}$ is the tangent space at $p \in \mathcal{M}$. Under this metric, the decision boundary between classes c and c' is the hyperplane $\{p \in \mathcal{M} : (w_c - w_{c'})^\top p = 0\}$, and each w_c defines a distance function $d_c(p) = w_c^\top p$ on $\mathcal{M}$. We stress that an individual weight w_c is not itself the metric: it is the differential (a covector) of the class-c score function, whose rank-one contribution to the metric is $w_c w_c^\top$, and the full metric $WW^\top$ is the sum of these contributions over classes. When below we say that the classifier "defines" or "induces" the metric, we mean precisely this: the metric object is $WW^\top$, and modifying a row w_c modifies the geometry by changing one of its rank-one components.

Given this geometric setup, the problem caused by sequential training becomes explicit. When the backbone updates from $f_{\theta_{t-1}}$ to f_{θ_t}, the manifold deforms ($\mathcal{M} \to \mathcal{M}'$). However, the classifier W

remains unchanged during this transition. Consequently, the pullback metric g_W is now evaluated on $\mathcal{M}'$ rather than $\mathcal{M}$, causing the decision boundaries to shift relative to the feature positions. The class prototypes μ_c (empirical means on $\mathcal{M}$) and μ'_c (means on $\mathcal{M}'$) diverge, and the alignment $w_c \approx \mu'_c/\|\mu'_c\|$ is broken. This metric drift accumulates across tasks as the manifold continues to deform, and is the root cause of classifier miscalibration in EFCIL.

E.2 Re-anchoring under Manifold Drift

We now analyze how manifold preservation (via FMC) and prototype correction (via KSC) jointly determine the effectiveness of PACR. Let μ_c^* be the true class center on the deformed manifold $\mathcal{M}'$, and let μ_c be the old prototype on the original manifold $\mathcal{M}$. FMC bounds the manifold drift: $\|\mu_c^* - \mu_c\| \leq \varepsilon$, where smaller ε indicates better manifold preservation. KSC estimates μ_c^* from the observed embedding shifts, yielding $\mu_c^{\text{compensated}}$ with error $\|\mu_c^{\text{compensated}} - \mu_c^*\| \leq \delta$. PACR then re-anchors via $w_c^{\text{new}} = (1-\beta)w_c^{t-1} + \beta \cdot \mu_c^{\text{compensated}}/\|\mu_c^{\text{compensated}}\|$.

At convergence of the initial training, the classifier weight aligns with the class center on $\mathcal{M}$: $w_c^{t-1} \approx \mu_c/\|\mu_c\|$. Under this condition and the cosine identity $\cos(u,v) = 1 - \frac{1}{2}\left\|u/\|u\| - v/\|v\|\right\|^2$, we obtain:

$$\cos(w_c^{t-1}, \mu_c^*) = 1 - O(\varepsilon^2), \tag{42}$$

$$\cos(\mu_c^{\text{compensated}}, \mu_c^*) = 1 - O(\delta^2). \tag{43}$$

The alignment of the re-anchored weight admits the lower bound:

$$\begin{aligned}\cos(w_c^{\text{new}}, \mu_c^*) \geq (1-\beta)\cos(w_c^{t-1}, \mu_c^*) \\ + \beta \cos(\mu_c^{\text{compensated}}, \mu_c^*).\end{aligned} \tag{44}$$

Substituting Eq. equation 42 and Eq. equation 43 into Eq. equation 44, we have:

$$\cos(w_c^{\text{new}}, \mu_c^*) \geq 1 - (1-\beta)\cdot O(\varepsilon^2) - \beta \cdot O(\delta^2). \tag{45}$$

Thus, Eq. equation 45 captures the complete causal chain of CAMP.

Eq. equation 45 reveals that without FMC, ε is large because the manifold deforms substantially across tasks. The term $(1-\beta)\cdot O(\varepsilon^2)$ then dominates the bound, and $\cos(w_c^{\text{new}}, \mu_c^*)$ can fall below $\cos(w_c^{t-1}, \mu_c^*)$, meaning re-anchoring degrades alignment. This matches our empirical finding that PACR alone underperforms the baseline.

With FMC, ε is small and the old weight remains well-aligned ($\cos(w_c^{t-1}, \mu_c^*) \approx 1$). With KSC, δ is small and the corrected prototype accurately points to μ_c^*. The bound then guarantees $\cos(w_c^{\text{new}}, \mu_c^*) \approx 1$, strictly improving alignment. This synergy is what drives the superior performance of the full CAMP pipeline.

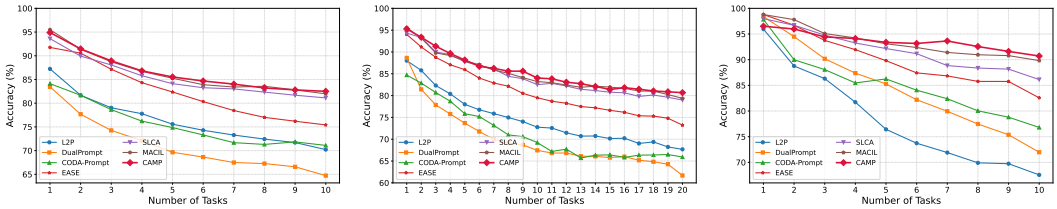
When FMC preserves the manifold (ε small), the old weight w_c^{t-1} is already close to optimal, contributing a near-1 term to the bound regardless of β. The role of PACR is merely to correct the residual drift captured by the $\beta \cdot O(\delta^2)$ term. A single small β ($= 0.3$) accomplishes this without overshooting toward the prototype direction. Setting $\beta = 1$ discards the w_c^{t-1} term entirely, losing the discriminative structure learned across tasks, which our experiments confirm degrades performance.

E.3 Connection to Neural Collapse

Finally, we note that PACR admits a natural interpretation through the lens of Neural Collapse (NC) (Papyan et al., 2020). At the terminal phase of training, NC theory predicts that $w_c \propto \mu_c$ (the so-called NC3 condition), meaning classifier weights converge to their class prototypes. Incremental training disrupts this alignment as the manifold deforms. PACR restores it by nudging w_c back toward the corrected prototype, without enforcing the full simplex ETF structure, which would overly constrain the representation for future classes.

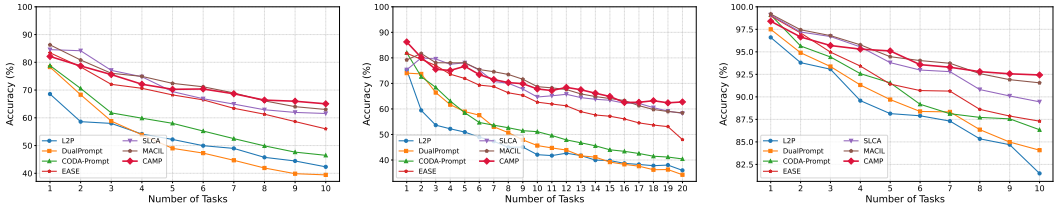
F Experimental Settings and Additional Results

In this appendix, we provide experimental settings and additional results.



(a) ImageNet-R (10 steps) (b) ImageNet-R (20 steps) (c) CUB-200 (10 steps)

Figure 3: Per-task accuracy curves for ImageNet-R (T=10, T=20) and CUB-200.



(a) ImageNet-A (10 steps) (b) ImageNet-A (20 steps) (c) CIFAR-100 (10 steps)

Figure 4: Per-task accuracy curves for ImageNet-A (T=10, T=20) and CIFAR-100.

F.1 Training Details

All experiments were conducted on a single NVIDIA GeForce RTX 4090 GPU (24 GB VRAM) and an Intel Core i9-13900KF CPU with 256 GB RAM, running Ubuntu 22.04 LTS. The implementation is based on PyTorch 2.0.1 with CUDA 11.8. Per-task accuracy curves are provided in Figure 3 and 4.

F.2 Resource Accounting

Table 6 reports the resource cost of each method. Storage counts only the method-specific parameters (prompts, adapters, or LoRA updates) and excludes the shared frozen backbone, as well as the prototype and covariance statistics, since every CA-based method in this comparison requires them and they are therefore a property of the evaluation protocol rather than of any single method. "Param" is the fraction of trainable parameters relative to the backbone, and storage is computed from it at float32. Training time is measured on CIFAR-100 with $T = 10$ on a single RTX 4090.

Table 6: Resource accounting. Trainable parameter fraction and storage are method-specific, excluding the shared frozen backbone and the prototype/covariance statistics common to all CA-based methods. Training time is measured on CIFAR-100 with $T = 10$ on a single RTX 4090. Entries marked "–" could not be recovered from our logs.

Method	**Param (%)**	**Train (min)**	**Inference (s)**	**Train storage (MB)**
L2P (Wang et al., 2022b)	0.2	22.4	27.2	0.31
DualPrompt (Wang et al., 2022a)	0.5	20.6	25.5	1.82
CODA-Prompt (Smith et al., 2023)	4.6	90.8	24.5	14.65
EASE (Zhou et al., 2024)	1.4	55.0	191.0	7.37
SLCA (Zhang et al., 2023)	100	89.5	60.9	343.2
MACIL (Wu et al., 2025)	1.0	77.2	101.7	4.75
SLDC (Rao et al., 2026)	3.5	23.4	14.4	16.52
CAMP (ours)	1.0	67.2	102.9	4.75

For CAMP, the only method-specific parameters are the LoRA updates on the key and value projections of all $L = 12$ blocks, giving $4drL = 4 \times 768 \times 32 \times 12 \approx 1.18$M trainable parameters, i.e. 4.75 MB in float32. At inference these updates are merged into the frozen backbone, so CAMP adds zero inference-time storage, the same deployment profile as the parameter-efficient baselines above.