



EFC++: Elastic Feature Consolidation with Prototype Re-balancing for Cold Start Exemplar-free Incremental Learning

Simone Magistri¹ · Tomaso Trinci² · Albin Soutif-Cormerais³ · Joost van de Weijer⁴ · Andrew D. Bagdanov¹

Received: 3 October 2025 / Accepted: 24 August 2026
© The Author(s) 2026

Abstract

Exemplar-free Class Incremental Learning (EFCIL) aims to learn from a sequence of tasks without having access to previous task data. In this paper, we consider the challenging Cold Start scenario in which insufficient data is available in the first task to learn a high-quality backbone. This is especially challenging for EFCIL since it requires high plasticity, resulting in feature drift which is difficult to compensate for in the exemplar-free setting. To address this problem, we propose an effective approach to consolidate feature representations by regularizing drift in directions highly relevant to previous tasks while employing prototypes to reduce task-recency bias. Our approach, which we call Elastic Feature Consolidation++ (EFC++), exploits a tractable second-order approximation of feature drift based on a proposed Empirical Feature Matrix (EFM). The EFM induces a pseudo-metric in feature space which we use to regularize feature drift in important directions and to update Gaussian prototypes. In addition, we introduce a post-training prototype re-balancing phase that updates classifiers to compensate for feature drift. This strategy allows to improve over our previous EFC method by mitigating the misalignment between stored prototypes and the evolving feature space. Extensive experimental results on Tiny-ImageNet, ImageNet-Subset, ImageNet-1K, and DomainNet show that EFC++ achieves a strong stability–plasticity trade-off in Cold Start and outperforms recent exemplar-free baselines. Code is available at https://github.com/simomagi/elastic_feature_consolidation.

Keywords Class-incremental learning · Continual learning · Catastrophic forgetting · Computer vision

1 Introduction

Deep Neural Networks (DNNs) like Convolutional Neural Networks (He et al., 2015) and Vision Transformers (Dosovitskiy et al., 2021) achieve strong performance across many

applications. In real-world applications like autonomous driving (Shaheen et al., 2022), multimedia forensics (Magistri et al., 2024a), healthcare (Lee and Lee, 2020) or remote sensing (Liu et al., 2022), data are often not stationary, necessitating the adaptation of DNNs to new data and tasks. Updating models is challenging due to the phenomenon known as *catastrophic forgetting* (McCloskey and Cohen, 1989; French, 1999), i.e., the tendency of DNNs to lose previously acquired knowledge when learning new tasks. Class Incremental Learning (CIL) addresses this problem by enabling models to learn new classes without retraining from scratch or storing all previous data (Verwimp et al., 2024; Masana et al., 2022). The key challenge in CIL is to find the right equilibrium between stability and plasticity during network training. Striking this balance is often referred to as the *stability-plasticity* dilemma (Wang et al., 2024; Mermillod et al., 2013).

Relying solely on data from the current task to mitigate catastrophic forgetting is challenging because the final deep representation and classifier tend to be highly biased

Communicated by Hong Liu.

Simone Magistri and Tomaso Trinci contributed equally to this work.

✉ Simone Magistri
simone.magistri@unifi.it

Tomaso Trinci
tomaso.trinci@unifi.it

¹ Media Integration and Communication Center (MICC), University of Florence, Florence, Italy

² Global Optimization Laboratory, University of Florence, Florence, Italy

³ LAMP Team, Computer Vision Center, Barcelona, Spain

⁴ LAMP Team, Computer Vision Center, Universitat Autònoma de Barcelona, Barcelona, Spain

towards the most recent task (a phenomenon known as *task-recency bias*). Additionally, the final classifier is never trained to distinguish among all the encountered classes (a phenomenon called *inter-task confusion* (Masana et al., 2022)). To address this, many recent CIL approaches, known as Exemplar-based Class Incremental Learning (EBCIL) methods, store samples from previous tasks (e.g., 20 per class) as exemplars and replay them while learning new tasks (Kang et al., 2022; Yan et al., 2021; Zhou et al., 2022). While effective, these methods require additional storage and computational resources, which can scale with the number of tasks (Trinci et al., 2025; Lanzillotta et al., 2026). Furthermore, storing samples can pose significant privacy issues, as it may not always be feasible to retain such data due to privacy concerns (Caselli et al., 2025; Qazi et al., 2026). For these reasons, in this paper we focus on the more general and challenging incremental learning setting without exemplars, a paradigm known as Exemplar-free Class Incremental Learning (EFCIL).

Elastic Weight Consolidation (EWC) is a pioneering work in EFCIL (Kirkpatrick et al., 2017). EWC addresses catastrophic forgetting via weight regularization. EWC is an elegant approach based on a Laplace approximation of the previous task posterior. When training on a new task, an approximate Fisher Information Matrix is used to regularize parameter drift to prevent the optimization from harming parameters important for previous tasks, while simultaneously increasing plasticity to learn new tasks. Since computing the full Fisher Information Matrix is intractable (its size is quadratic in the number of network parameters), EWC uses a diagonal approximation of the matrix. This approximation is an aggressive simplification compared to the full matrix and results in a significant performance drop (Liu et al., 2018; Ritter et al., 2018).

Given the difficulties in regularizing network weights, recent EFCIL methods rely on the more tractable functional regularization paradigm which mitigates network activation drift during incremental learning. Feature distillation, consisting in regularizing feature space drift via ℓ_2 regularization, is a central component in recent state-of-the-art EFCIL approaches (Liu et al., 2020; Zhu et al., 2021b, 2022; Toldo and Ozay, 2022; Yu et al., 2020). Feature distillation is often combined with class prototypes, either learned (Toldo and Ozay, 2022) or based on class means (Zhu et al., 2021b), to perform pseudo-rehearsal of features from previous tasks in order to reduce task-recency bias and inter-task confusion phenomenon. Prototypes – differently from exemplars – offer an efficient and privacy-preserving way to mitigate forgetting.

While feature distillation effectively mitigates forgetting through increased stability, it does so at the cost of reduced plasticity for learning new tasks (Douillard et al., 2020). A similar emphasis on stability is also observed in other recent

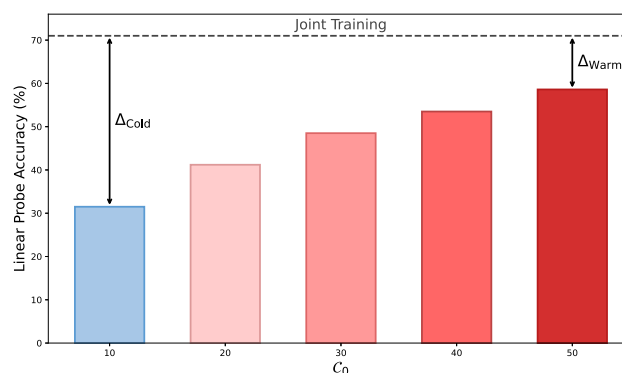


Fig. 1 Plasticity potential in Cold and Warm Start. We train a ResNet-18 on C_0 classes of CIFAR-100 for $C_0 = 10, 20, \dots, 50$, and evaluate feature quality via linear probing (Davari et al., 2022) on all 100 classes. The plasticity potential Δ , defined as the maximum performance gain which can be obtained with a plastic versus a frozen backbone, is quantified as the performance gap between Joint Training and a model frozen on a subset of classes. Note that Δ_{Cold} (the *Cold Start* scenario) is significantly larger than Δ_{Warm} (the *Warm Start* case). This is due to the inability to learn a strong feature extractor on only C_0 classes and hence greater plasticity is required to incrementally learn new classes. Freezing the backbone in such settings limits adaptability and ultimately constrains performance on subsequent tasks

EFCIL methods that freeze the backbone after the first task and focus solely on incrementally learning the classifier (Petit et al., 2023; Goswami et al., 2023). This emphasis on stability largely stems from the standard evaluation setting for EFCIL methods (Zhu et al., 2022, 2021b; Toldo and Ozay, 2022; Petit et al., 2023; Goswami et al., 2023). In this setting, called the *Warm Start* scenario, the first task includes a large fraction of the total classes, typically 40% or 50%, making stability more critical than plasticity. Since a high-quality feature extractor is learned in the first task, it improves generalization in subsequent tasks. However, when the initial task is smaller, an incremental learning method must increase plasticity to compensate for the lack of a strong initial feature extractor, requiring substantial updates to the backbone in order to effectively learn new tasks.

In Figure 1 we give an empirical analysis via linear probing to demonstrate the importance of plasticity in EFCIL for different initial task sizes. These results show that for smaller initial task sizes (e.g. $C_0 = 10$), the plasticity potential Δ – defined as the maximum performance gain which can be obtained with a plastic versus a frozen backbone – is highest. This reflects the greater need for feature adaptation in later incremental learning stages, as a weaker initial backbone demands more flexibility to incorporate new knowledge.

In this paper we examine EFCIL in the more challenging *Cold Start* scenario in which the first task is insufficiently large to learn a high-quality backbone and methods must be plastic and adapt their backbone to new tasks. Although this scenario has been explored in the EBCIL literature (Zhou et al., 2022; Yan et al., 2021), it is rarely addressed in EFCIL

due to several challenges. In Cold Start scenarios, representations are more prone to change than in Warm Start, making it difficult to control drift without compromising plasticity – especially when there are no exemplars to rely on. Additionally, using a strict regularizer, like feature distillation or freezing the backbone, prevents the network from adapting to new data. Therefore, an alternative to feature distillation is necessary and an exemplar-free mechanism is needed to adapt previous-task classifiers to the evolving backbone.

To address the challenges of Cold Start, we propose a novel EFCIL approach, called *Elastic Feature Consolidation with Prototype Re-balancing* (EFC++), that builds upon our previous method EFC (Magistri et al., 2024b). We introduce the Empirical Feature Matrix (EFM), a tractable second-order pseudo metric that regularizes changes along directions most relevant to past tasks while promoting the plasticity needed in Cold Start. Unlike the Fisher Information Matrix, the EFM can be easily computed and stored, since its size depends only on the feature space dimensionality. As in EFC, EFC++ also relies on Gaussian prototypes to mitigate inter-task confusion. However, we show that in high-plasticity Cold Start regimes, the joint optimization of the feature extractor and prototype-rehearsal based classifier used in EFC makes stored prototypes progressively misaligned with the evolving feature space. EFC++ addresses this limitation by decoupling feature learning from classifier adaptation: features are learned with EFM regularization, while prototypes are updated after feature learning and used to re-balance a global classifier over all seen classes. This design mitigates prototype-feature misalignment and enables a more effective use of EFM regularization. In addition, compared to our previous work (Magistri et al., 2024b), we perform a more detailed diagnostic study of the method, including Warm versus Cold Start drift behavior, prototype re-balancing ablations, spectral and perturbation analyses of the EFM, hyperparameter sensitivity, efficiency considerations, and remaining failure modes, including the central role of class mean estimation over covariance estimation in EFCIL prototype rehearsal.

Contributions. Our main contributions are:

- **Cold Start EFCIL.** We study exemplar-free class-incremental learning under the challenging *Cold Start* setting, in which the first task is insufficient to learn a strong backbone. We show that the plasticity required in this regime induces structured feature drift, disproportionately affects earlier tasks, and progressively degrades prototype reliability.
- **Empirical Feature Matrix.** We present the Empirical Feature Matrix (EFM), a tractable feature-space analogue of second-order importance estimation. The EFM induces a pseudo-metric that selectively regularizes drift

along directions important for previous tasks, while preserving plasticity in less relevant directions.

- **Elastic Feature Consolidation.** We propose Elastic Feature Consolidation (EFC) and its extension *Elastic Feature Consolidation with Prototype Re-balancing* (EFC++). EFC combines EFM-based regularization with asymmetric prototype rehearsal and EFM-guided prototype drift compensation. EFC++ further decouples feature learning from classifier adaptation through prototype re-balancing, improving prototype alignment in Cold Start scenarios.
- **Comprehensive Evaluation.** We evaluate EFC and EFC++ on CIFAR-100, Tiny ImageNet, ImageNet Subset, ImageNet-1K, and DomainNet in *Cold Start* and *Warm Start* regimes. Results show competitive performance on CIFAR-100, and state-of-the-art performance on more challenging benchmarks compared to recent EFCIL methods.

The paper is organized as follows. Section 3 introduces the Empirical Feature Matrix (EFM). Section 4 presents Elastic Feature Consolidation with Prototype Re-balancing (EFC++). Sections 5–6 present the experimental setup and comparisons with state-of-the-art EFCIL methods. Section 7 reports ablations and additional analyses. Finally, Section 8 concludes with our main contributions and future directions.

2 Related Work

The primary goal of incremental learning is to continuously acquire new knowledge from a non-stationary data stream while mitigating the issue of catastrophic forgetting. In this study, we focus on *offline class incremental* learning problems. The class-incremental scenario is distinguished from the task-incremental one in that no task information is available at test time and thus is more challenging (Van De Ven et al., 2022; De Lange et al., 2022). Offline incremental learning allows multiple passes on task datasets during training, while online incremental learning considers continuous data streams in which each sample is accessed just once during training (Lopez-Paz and Ranzato, 2017; Aljundi et al., 2019a; Zhang et al., 2022; Soutif-Cormerais et al., 2023; Aljundi et al., 2019b). Recently, more general lifelong learning formulations have also been proposed beyond the class-incremental setting considered in this work (Vogelstein et al., 2025), including continual learning for vision-brain understanding across subjects (Nguyen et al., 2026).

A significant portion of offline class incremental learning research focuses on Exemplar-Based Class Incremental Learning (EBCIL) approaches. These methods integrate regularization techniques with replay mechanisms, utilizing a memory of previous class samples that are replayed dur-

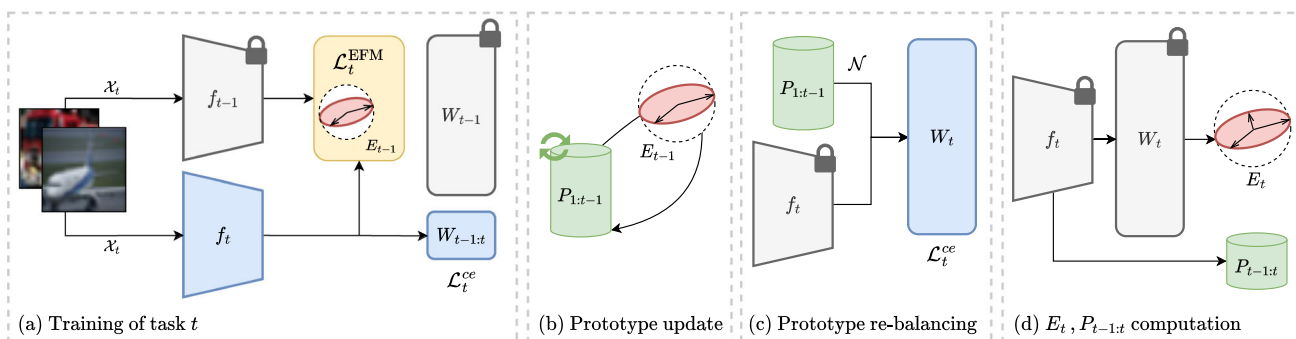


Fig. 2 Elastic Feature Consolidation with Prototype Re-balancing (EFC++). **(a)** EFC++ leverages the Empirical Feature Matrix (EFM) to mitigate drift in feature representations by identifying important directions for previous tasks to reduce forgetting while enhancing plasticity for learning new tasks (Section 3.3). In this phase, the feature extractor f_t and the current task classifier with weights $W_{t-1:t}$ are trained with EFM regularization and cross-entropy loss (Section 4.3) **(b)** After

training, EFC++ uses the EFM to update the prototypes of previous task classes based on the drift induced by the most recent task (Section 4.4). **(c)** EFC++ uses Gaussian prototypes, together with current task features, for training previous and current task classifiers with weights $W_t = [W_{t-1}, W_{t-1:t}]$ via a prototype re-balancing phase (Section 4.5). **(d)** Before training on the next task, the new EFM and the prototypes of the current task classes are computed

ing the learning of new tasks. iCaRL (Rebuffi et al., 2017) employs Nearest Mean Classification along with exemplars and uses Knowledge Distillation (Hinton et al., 2015) during training to mitigate forgetting. Techniques like BiC (Wu et al., 2019) and IL2M (Belouadah and Popescu, 2019) correct network outputs using exemplars to mitigate inter-task confusion and task-recency bias. DER++ (Buzzega et al., 2020) and its recent variant X-DER (Boschini et al., 2023) replay both logits and ground-truth labels via self-distillation on exemplars. LUCIR (Hou et al., 2019), PODNet (Douillard et al., 2020), and AFC (Kang et al., 2022) combine cosine normalization and distillation losses with exemplars to reduce forgetting. Another category of exemplar-based approaches focuses on expanding sub-network structures across incremental steps. AANET (Liu et al., 2021) adds two residual blocks to mask layers, balancing stability and plasticity. DER (Yan et al., 2021) trains a single backbone for each incremental task, while MEMO (Zhou et al., 2022) shares generalized residual blocks and extends specialized blocks for new tasks to improve performance and efficiency.

Recent exemplar-based works further analyze the role of replay scheduling, buffer size, and rehearsal strategies from both empirical and theoretical perspectives (Kang et al., 2025; Lanzillotta et al., 2026; Deng et al., 2025). This research has also expanded to more challenging areas like the video domain. For instance Jiang et al. (2026), develop a Teacher Agent approach that minimizes rehearsal overhead by substituting the teacher network with a lightweight agent that generates soft labels directly. The main drawbacks of exemplar-based approaches are the lack of privacy preservation and high computational and storage costs, especially when using a growing memory buffer or when dynamically expanding the network.

Exemplar-free Class Incremental Learning (EFCIL) mitigates catastrophic forgetting without relying upon exemplars from previous tasks. This setting has also been explored in application-specific domains such as traffic sign classification, where storing past samples may be undesirable in deployed systems (Liu et al., 2026). Early works such as EWC (Kirkpatrick et al., 2017; Huszár, 2018), Rwalk (Chaudhry et al., 2018), MAS (Aljundi et al., 2018), SI (Zenke et al., 2017), focused on calculating importance scores for all neural network weights. These importance scores are then used to regularize the network training, ensuring that the current learned weights do not update parameters crucial to the performance of previous task. Benzing (2022) show that the importance score computed by the aforementioned methods approximates the Fisher Information Matrix. Recently, Liu and Chang (2026) showed that modifying the logits during the computation of the Fisher Information Matrix can improve EWC weight regularization by producing more reliable parameter importance estimates.

Another research direction, exemplified by methods like Learning without Forgetting (LwF) (Li and Hoiem, 2017) and Less-forgetful Learning (LfL) (Jung et al., 2016), utilizes functional regularization techniques such as knowledge and feature distillation respectively. These methods constrain network activations to mirror those of a previously trained network, a strategy that has been shown to surpass weight regularization in performance (Masana et al., 2022). Most recent functional regularization approaches use feature distillation (Zhu et al., 2021a; Yu et al., 2020; Zhu et al., 2021b, 2022; Toldo and Ozay, 2022; Liu et al., 2020), or even freeze the feature extractor after the first task (Petit et al., 2023; Goswami et al., 2023; Belouadah and Popescu, 2018; He et al., 2026), greatly reducing the plasticity of the network. For this reason, they predominantly consider Warm Start scenar-

ios in which the first task is much larger than the others, so that reducing plasticity after the first task has less impact on the performance.

On top of this decrease in plasticity, merely using functional regularization is not enough to learn new boundaries between classes from previous tasks and classes from new ones. To address this issue without using exemplars, a line of research employs generative models (Liu et al., 2020; Shin et al., 2017; Xiang et al., 2019; Smith et al., 2021; Ayub and Wagner, 2021; Gao et al., 2022; Wang et al., 2025) to create synthetic samples from previous tasks and replay them while learning the new tasks. However, training generative models concurrently with new tasks is inefficient, and these models are prone to forgetting (Zhu et al., 2021b). Consequently, recent approaches combine regularization with *prototype rehearsal* (Zhu et al., 2021b, a). Prototypes are feature space statistics (typically class means) used to reinforce the boundaries of previous-task classes without the need to preserve exemplars. PASS (Zhu et al., 2021b) and IL2A (Zhu et al., 2021a) propose the integration of Gaussian prototypes and self-supervised learning to learn transferable features for future tasks. Evanescent (Toldo and Ozay, 2022) learns and updates prototype representations during incremental learning. SSRE (Zhu et al., 2022) replays over-sampled prototypes during the current task training and employs a strategy to reorganize the network structure to transfer invariant knowledge across tasks. FeTrIL (Petit et al., 2023) fixes the feature extractor after the initial large task and generates pseudo-samples, combining prototypes with current task features, to train a linear model that discriminates all observed classes. Similarly, FeCAM (Goswami et al., 2023) freezes the backbone after the first task, but uses class prototypes and covariances to classify samples with the Mahalanobis distance.

In our previous work, Elastic Feature Consolidation (EFC) (Magistri et al., 2024b), we revisited the *Cold Start* scenario for EFCIL, showing the need for greater plasticity throughout incremental learning. Since then, recent works have addressed this setting through drift compensation (Gomez-Villa et al., 2025; He et al., 2025), old-class recovery from current task data (Goswami et al., 2024), or class statistics adaptation under feature drift (Rypešć et al., 2024). In this extension of our previous work we further analyze the challenges of Cold Start and the limitations of EFC, and propose EFC++, a training strategy that better couples representation learning and classifier adaptation in regimes requiring high plasticity.

3 Feature Drift Regularization via the Empirical Feature Matrix

In this section, we introduce the general EFCIL framework and then derive a pseudo-metric in feature space, induced by a positive semi-definite matrix we call the *Empirical Feature Matrix* (EFM), used as a regularizer to control feature drift during class-incremental learning.

3.1 Exemplar-free Class-Incremental Learning (EFCIL)

During class-incremental learning, a model $\mathcal{M}_t$ is sequentially trained on K tasks, each characterized by a disjoint set of classes $\{\mathcal{C}_t\}_{t=1}^K$, where each $\mathcal{C}_t$ is the set of labels associated with the task t . We denote the incremental dataset as $\mathcal{D} = \{\mathcal{X}_t, \mathcal{Y}_t\}_{t=1}^K$, where $\mathcal{X}_t$ and $\mathcal{Y}_t$ are, respectively, the set of samples and the set of labels for task t . The incremental model $\mathcal{M}_t$ at task t consists of a feature extraction backbone $f(\cdot; \theta_t)$ shared across all tasks and whose parameters θ_t are updated during training, and a classifier $W_t \in \mathbb{R}^{n \times \sum_{j=1}^t |\mathcal{C}_j|}$ which grows with each new task. Let σ be the softmax activation function, the model output at task t is the composition of feature extractor and classifier:

$$\mathcal{M}_t(x; \theta_t, W_t) \equiv p(y|x; \theta_t, W_t) = \sigma(W_t^\top f(x; \theta_t)). \quad (1)$$

For notation purposes, we define $\mathbf{X} \sim \mathcal{X}_t$ as a batch of current task data with labels $\mathbf{Y} \sim \mathcal{Y}_t$, sampled from the overall dataset, and $f(\mathbf{X}; \theta_t)$ as the feature extractor output on the mini-batch. Thus, the model output at task t for the mini-batch $\mathbf{X}$ is:

$$\mathcal{M}_t(\mathbf{X}; \theta_t, W_t) \equiv p(\mathbf{Y}|\mathbf{X}; \theta_t, W_t) = \sigma(W_t^\top f(\mathbf{X}; \theta_t)). \quad (2)$$

Directly training $\mathcal{M}_t$ (i.e. fine-tuning) with a cross-entropy loss at each task t results in backbone drift because at task t we see no examples from previous-task classes. This progressively invalidates the previous-task classifiers and results in forgetting. This issue is closely related to feature forgetting in continually learned representations (Hess et al., 2024). In addition to the cross-entropy loss, many state-of-the-art EFCIL approaches use a regularization loss to reduce backbone drift, and a prototype loss to adapt previous-task classifiers to new features.

3.2 Weight Regularization and Feature Distillation

Kirkpatrick et al. (2017) showed that using ℓ_2 regularization to constrain weight drift reduces forgetting but does not leave enough plasticity for learning new tasks. Hence, they proposed Elastic Weight Consolidation, which relaxes the ℓ_2 constraint with a quadratic constraint based on a diagonal

approximation of the Empirical Fisher Information Matrix (E-FIM):

$$F_t = \mathbb{E}_{x \sim \mathcal{X}_t} \left[\mathbb{E}_{y \sim p(y|x; \theta_t^*)} \left[\left(\frac{\partial \log p(y)}{\partial \theta_t^*} \right) \left(\frac{\partial \log p(y)}{\partial \theta_t^*} \right)^\top \right] \right], \quad (3)$$

where θ_t^* is the model trained after task t . The E-FIM induces a pseudo-metric in parameter space (Amari, 1998) encouraging parameters to stay in a low-error region for previous-task models:

$$\mathcal{L}_t^{\text{E-FIM}} := \lambda_{\text{E-FIM}} (\theta_t - \theta_{t-1}^*)^\top F_{t-1} (\theta_t - \theta_{t-1}^*). \quad (4)$$

The main limitation of approaches of this type is the need for approximations of the E-FIM (e.g. a diagonal assumption). This makes the computation of the E-FIM tractable, but in practice is unrealistic since it does not consider off-diagonal entries that represent influences of interactions between weights on the log-likelihood. To overcome these limitations, more recent EFCIL approaches (Zhu et al., 2021b, 2022) rely on feature distillation (FD), which scales better in the number of parameters:

$$\mathcal{L}_t^{\text{FD}} := \sum_{x \in \mathbf{X}} \|f(x; \theta_t) - f(x; \theta_{t-1})\|_2, \quad (5)$$

where $\mathbf{X} \sim \mathcal{X}_t$.

The isotropic regularizer using the ℓ_2 distance is too harsh a constraint for learning new tasks (Douillard et al., 2020). We propose to use a pseudo-metric in *feature space*, induced by a matrix which we call the *Empirical Feature Matrix* (EFM), which constrains directions in *feature space* most important for previous tasks, while allowing more plasticity in other directions when learning new tasks. As shown in Figure 1, increasing plasticity is particularly relevant in Cold Start incremental learning since, in the initial task, a high-quality feature representation cannot be learned and methods must be plastic to adapt their backbone to new tasks.

3.3 The Empirical Feature Matrix

Our goal is to regularize feature drift, but we do not want to do so isotropically as in feature distillation. Instead, we draw inspiration from how the E-FIM identifies important directions in parameter space and take a similar approach in feature space (see Figure 2a).

The local and empirical feature matrices. Let $f(x; \theta_t)$ be the feature vector extracted from input $x \in \mathcal{X}_t$ from task t and $p(y) = p(y|f(x; \theta_t); W_t)$ the discrete probability distribution over all the classes seen so far. We define the *local feature matrix* as:

$$E_{f(x; \theta_t)} = \mathbb{E}_{y \sim p(y)} \left[\left(\frac{\partial \log p(y)}{\partial f(x; \theta_t)} \right) \left(\frac{\partial \log p(y)}{\partial f(x; \theta_t)} \right)^\top \right]. \quad (6)$$

The Empirical Feature Matrix (EFM) associated with task t is obtained by taking the expected value of Eq. 6 over the entire dataset at task t :

$$E_t = \mathbb{E}_{x \sim \mathcal{X}_t} [E_{f(x; \theta_t)}]. \quad (7)$$

Both $E_{f(x; \theta_t)}$ and E_t are symmetric, as they are weighted sums of symmetric matrices, and positive semi-definite. Naively computing $E_{f(x; \theta_t)}$ requires a complete forward pass and a backward pass up to the feature embedding, however we exploit an analytic formulation that avoids computing gradients:

$$E_{f(x; \theta_t)} = \mathbb{E}_{y \sim p(y)} [W_t (I_m - P)_y (W_t (I_m - P)_y)^\top], \quad (8)$$

where $m = \sum_{j=1}^t |C_j|$, I_m the identity matrix of dimension $m \times m$, and P is a matrix built from the row of softmax outputs associated with $f(x; \theta_t)$ replicated m times (see Appendix B for a derivation).

To gain additional insight into what this matrix measures, we highlight some (information) geometry aspects of E_t . The Empirical Fisher Information Matrix provides information about the information geometry of parameter space. In particular, it can be obtained as the second derivative of the KL-divergence for small weight perturbations (Martens, 2020; Gementieri and Fioresi, 2022). Similarly, the Empirical Feature Matrix provides information about the information geometry of *feature space*. Since $E_{f_t(x)}$ is positive semi-definite, we can interpret it as a pseudo-metric in feature space and use it to understand which perturbations δ of features most affect the predicted probability:

$$\begin{aligned} & \text{KL}(p(y | f(x; \theta_t) + \delta; W_t) || p(y | f(x; \theta_t); W_t)) \\ &= \delta^\top E_{f(x; \theta_t)} \delta + \mathcal{O}(\|\delta\|^3). \end{aligned} \quad (9)$$

The Empirical Feature Matrix loss. We now have all the ingredients for our regularization loss. Let $\delta(x) = f(x; \theta_t) - f(x; \theta_{t-1})$ be the drift between the features of sample x extracted from the current model $\mathcal{M}_t$ and the model $\mathcal{M}_{t-1}$ trained on the previous task, then

$$\mathcal{L}_t^{\text{EFM}} := \mathbb{E}_{x \in \mathbf{X}} \left[\delta(x)^\top (\lambda_{\text{EFM}} E_{t-1} + \eta I) \delta(x) \right], \quad (10)$$

where $\mathbf{X} \sim \mathcal{X}_t$. In Eq. 10 we use a damping term $\eta \in \mathbb{R}$ to constrain features to stay in a region where the quadratic second-order approximation of the KL-divergence of the log-likelihood in Eq. 9 remains valid (Martens and Sutskever, 2012). However, to ensure that our regularizer does not degenerate into feature distillation, we must ensure that $\lambda_{\text{EFM}} v_i > \eta$, where v_i are the non-zero eigenvalues of the

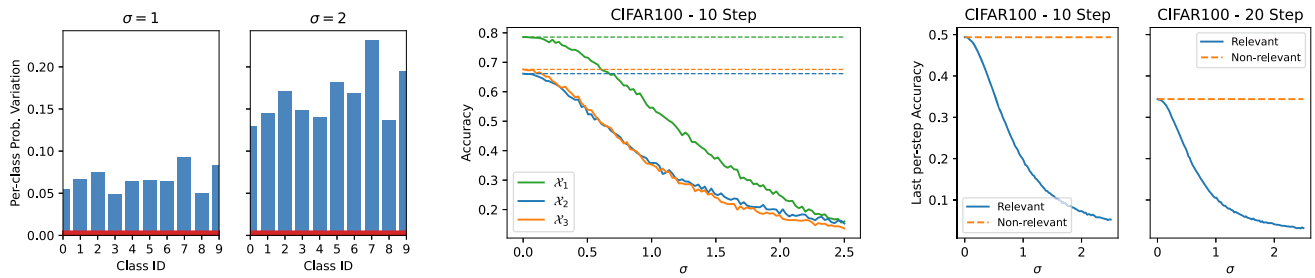


Fig. 3 The regularizing effects of E_t on the Cold Start CIFAR-100 - 10 and 20 step scenarios (see Section 5 for details on dataset settings). **Left:** Perturbing features in the principal directions of E_1 results in significant changes in classifier outputs (in blue), while perturbations in non-principal directions leave the outputs unchanged (in red). **Middle:** If we continue incremental learning up through task 3 and perturb features from all three tasks in the principal (solid lines) and non-principal

(dashed lines) directions of E_3 , we see that E_3 captures *all* important directions in feature space up through task 3. **Right:** At the end of training, we observe the same behavior: the last per-step accuracy (see Eq. 21), representing the average accuracy over all tasks after the last training session, decreases only when perturbed in directions of E_{10} or E_{20} relevant for previous tasks in the 10-step and 20-step scenarios, respectively

EFM. Additional analysis of these constraints and the spectrum of the EFM are given in Section 7.3.

It should be noted that the regularization in Eq. 10 does not explicitly suppress perturbations along the principal directions of E_{t-1} , but rather discourages movement along them. The term $\delta(x)^\top E_{t-1} \delta(x) = \delta(x)^\top U_{t-1} \Lambda_{t-1} U_{t-1}^\top \delta(x)$, highlights this, where U_{t-1} is the eigenvector matrix of E_{t-1} and Λ_{t-1} is the corresponding diagonal matrix of eigenvalues sorted from larger to smaller. The penalty is larger when $\delta(x)$ aligns with the eigenvectors of E_{t-1} associated with larger eigenvalues in Λ_{t-1} . Conversely, the loss is lower when $\delta(x)$ aligns with eigenvectors corresponding to smaller eigenvalues. As a result, updates are naturally biased toward directions of lower significance since they incur a smaller penalty. The damping term $\eta \delta(x)^\top I \delta(x)$ stabilizes updates and limits drift in the directions associated with lower eigenvalues. In the next section we give empirical evidence of the effect of this elastic regularization.

The Regularizing effects of the EFM. By definition, eigenvectors of the EFM associated with strictly positive eigenvalues correspond to feature directions where perturbation most significantly impact the predictions. To empirically validate this, we perturb features along these *principal* directions and measure the variation in probability outputs.

Let $f_t = f(x; \theta_t) \in \mathbb{R}^n$ be the feature vector extracted from x , and E_t denote the EFM after training on task t . We define a perturbation vector $\varepsilon_{1:k} \in \mathbb{R}^n$, where the first k entries are sampled from a Gaussian $\mathcal{N}(0, \sigma)$, with k being the number of strictly positive eigenvalues of the spectral decomposition of E_t . The remaining $n-k$ entries are set to zero. Then, letting U_t be the matrix whose columns are the eigenvectors of E_t , we compute the perturbed features vector $\tilde{f}_t$ as follows:

$$\tilde{f}_t = f + U_t^\top \varepsilon_{1:k}. \quad (11)$$

In Figure 3 (left), perturbing the feature space along these principal directions after training the first task, leads to a substantial variation in class probabilities. Conversely, applying the complementary perturbation $\varepsilon_{k:n}$, i.e., setting zero on the first k directions and applying Gaussian noise on the remaining $n-k$ directions, has no impact on the output. In Figure 3 (middle) we see that, after training three tasks, perturbing in the principal directions of E_3 degrades performance across all tasks, while perturbations in non-principal directions continue to have no effect on the accuracy. This shows that E_3 captures the important feature directions for all previous tasks and that regularizing drift in these directions should mitigate forgetting.

Finally, the rightmost plot in Figure 3 illustrates the per-step accuracy at the end of the training, reflecting the overall continual learning performance (see Section 5 for a formal definition). As observed earlier, perturbing features along the principal directions of E_{10} (or E_{20}) rapidly decreases performance, while perturbations in non-principal directions have no impact on the overall accuracy. This confirms that this behavior remains consistent across all tasks.

4 Elastic Feature Consolidation with Prototype Re-balancing

Regularizing the feature space via the EFM in combination with a classification loss on current task classes is not sufficient by itself in exemplar-free class incremental learning. Although EFM limits feature drift along directions important for previous tasks, the final classifier must still discriminate among all the classes observed so far. This is challenging because the classifier W_{t-1} is extended to include the new $|\mathcal{C}_t|$ classes, yielding $W_t = [W_{t-1}, W_{t-1:t}]$. Trained only on current task samples, it cannot separate old and new classes, leading to *inter-task confusion* and *task-recency*

bias (Masana et al., 2022; Zhuang et al., 2022; Wu et al., 2019).

In this section, we introduce the notation for Gaussian prototype rehearsal, which we use for classifier adaptation between old and new classes. We revisit the PR-ACE loss in EFC and discuss why coupling prototype rehearsal with feature extractor training can be problematic in the high-plasticity Cold Start regime. Finally, we introduce Elastic Feature Consolidation with Prototype Re-balancing (EFC++), which decouples feature learning from inter-task classifier adaptation through a post-training prototype re-balancing phase.

4.1 Gaussian Prototypes for Classifier Adaptation

Let $\mathcal{C}_t$ be the set of classes introduced at task t , and let $\mathcal{C}_{1:t-1} := \bigcup_{s=1}^{t-1} \mathcal{C}_s$ denote the set of classes observed before task t . For each previous class $c \in \mathcal{C}_{1:t-1}$, we denote by p_{t-1}^c the prototype stored before learning task t . When a class $c \in \mathcal{C}_s$ is first introduced at task s , with $s \in \{1, \dots, t-1\}$ its prototype is initialized as the empirical mean of its deep features (Zhu et al., 2021b), while its corresponding covariance Σ^c is computed from the same features (Liu et al., 2020). The resulting class-level statistics are retained across incremental steps in the prototype buffer $P_{1:t-1}$.

During classifier adaptation at task t , we rehearse previous classes by sampling Gaussian-perturbed prototypes $\tilde{p}_{t-1}^c \sim \mathcal{N}(p_{t-1}^c, \Sigma^c)$. These Gaussian prototypes provide synthetic samples at feature level for previous classes and allow the classifier weights W_t to be trained over old and current classes *without* storing real images.

4.2 EFC and the Limitations of PR-ACE in Cold Start

Elastic Feature Consolidation (EFC) (Magistri et al., 2024b) combines the empirical feature matrix loss $\mathcal{L}_t^{\text{EFM}}$ (Eq. 10), inhibiting drift in important directions for previous tasks, with the *Asymmetric Prototype Replay* (PR-ACE) loss:

$$\mathcal{L}_t^{\text{EFC}} := \mathcal{L}_t^{\text{EFM}} + \mathcal{L}_t^{\text{PR-ACE}}, \quad (12)$$

where

$$\mathcal{L}_t^{\text{PR-ACE}} := \mathcal{L}_t^{\text{ce}}(f(\mathbf{X}, \theta_t); W_{t-1:t}) + \mathcal{L}_t^{\text{ce}}(\tilde{\mathbf{P}} \cup f(\hat{\mathbf{X}}, \theta_t); W_t). \quad (13)$$

Here $\mathbf{X}, \hat{\mathbf{X}} \sim \mathcal{X}_t$ are two distinct batches of current task data, while $\tilde{\mathbf{P}}$ is a batch of augmented prototypes, where each prototype $\tilde{p}_{t-1}^c \in \tilde{\mathbf{P}}$ is sampled from the Gaussian $\mathcal{N}(p_{t-1}^c, \Sigma^c)$. The term $\mathcal{L}_t^{\text{ce}}(f(\mathbf{X}, \theta_t); W_{t-1:t})$ is a cross-entropy loss restricted to the current task classes, while the term with prototypes $\mathcal{L}_t^{\text{ce}}(\tilde{\mathbf{P}} \cup f(\hat{\mathbf{X}}, \theta_t); W_t)$ affects old and

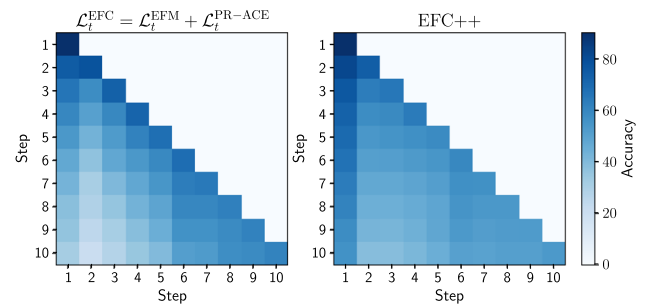


Fig. 4 Accuracy after each incremental step on the Cold Start CIFAR-100 10-step scenario. **Left:** In EFC, which combines EFM regularization with the asymmetric PR-ACE loss to balance current task data with prototypes during training, older tasks are forgotten more quickly than more recent ones. **Right:** EFC++, which applies EFM regularization during backbone training and a post-training prototype re-balancing phase, achieves a better plasticity-stability trade-off

current task classifier weights $W_t = [W_{t-1}, W_{t-1:t}]$. PR-ACE samples prototypes and images uniformly over all seen classes.

After each task, EFC updates the stored prototypes with the EFM to partially compensate for backbone drift and computes the prototypes, covariances, and EFM used in subsequent tasks; see Appendix C for the full EFC algorithm. We detail the EFM-based prototype update in Section 4.4, where it also becomes a key component of EFC++.

Limitations of PR-ACE in Cold Start.

We identify limitations in using PR-ACE for training the final classifier. Since PR-ACE is applied during feature extractor training, the stored prototypes for previous classes gradually lose their representativeness of the true class means over the course of current task epochs. This occurs because the feature space itself is evolving over time as a result of optimization, leading to a shift in the alignment between stored prototypes and their actual class means. Consequently, the classifier is trained with prototypes that increasingly deviate from the actual class means.

This misalignment is further compounded by the cross-entropy term $\mathcal{L}_t^{\text{ce}}(\tilde{\mathbf{P}} \cup f(\hat{\mathbf{X}}, \theta_t); W_t)$ on the right-hand side of Equation 13, which induces an *additional drift in the feature representation* along the relevant directions for previous tasks identified by the EFM. This drift arises from the backward pass on $\hat{\mathbf{X}}$, which updates the weights θ_t and alters the feature representations. Because PR-ACE uniformly samples prototypes $\tilde{\mathbf{P}}$ and new data $\hat{\mathbf{X}}$ based on class labels, mini-batches contain a larger proportion of current task data in earlier tasks compared to later tasks. Consequently, classes introduced in earlier incremental steps undergo stronger feature drift during their own training stage (see Figure 5 for drift analysis).

For illustration, consider the Cold Start scenario where each task introduces an equal number of classes. In the second task, PR-ACE forms batches containing 50% prototypes

and 50% image data; in the third task, this shifts to two-thirds prototypes and one-third current data. Because the cross-entropy loss is averaged over both prototypes and new data, the gradient updates on the feature extractor become less severe for later tasks. As more tasks are added, the drift in the feature representation along the relevant directions of the EFM diminishes. Consequently, prototypes from earlier tasks remain farther from the actual class means than those from more recent tasks, leading to forgetting. This effect is illustrated in Figure 4 (left), where in EFC the performance on the first five tasks declines faster than on the last five.

Managing the drift caused by PR-ACE is particularly crucial in the Cold Start scenario, where, during initial training, a strong backbone cannot be learned and thus representations are more susceptible to change compared to the Warm Start regime (see Section 7.2). As we will show in the next section, EFC++ mitigates this issue by removing prototype replay from feature extractor training and postponing inter-task classifier adaptation to a separate post-training phase.

4.3 Feature Learning without Prototype Rehearsal

We decouple the regularization phase via the Empirical Feature Matrix and the phase of learning the final inter-task classifier. In the first stage (Figure 2a), we train the *feature extractor* and the *current task classifier* with the following training loss:

$$\mathcal{L}_t^{\text{train}} := \mathcal{L}_t^{\text{EFM}} + \mathcal{L}_t^{\text{ce}}(f(\mathbf{X}, \theta_t); W_{t-1:t}). \quad (14)$$

where $\mathbf{X} \sim \mathcal{X}_t$ and $W_{t-1:t}$ represents the weights of the classifier associated with the latest classes. The cross-entropy loss is evaluated only on the current task classes, and is used to learn the features of the current task. $\mathcal{L}_t^{\text{EFM}}$ is the regularization loss based on Empirical Feature Matrix (see Eq. 10) and is used to mitigate feature drift in the relevant directions for the previous task while allowing plasticity in the other directions.

4.4 Prototype Drift Compensation via the EFM

After feature learning, the backbone has changed from θ_{t-1} to θ_t . Therefore, the prototypes stored before learning task t may no longer be aligned with the current feature space. Yu et al. (2020) showed that the drift in *embedding networks* can be estimated using current task data. They suggest estimating the drift of class means from previous-tasks by looking at feature drift in closely related classes from the current task. They leverage this drift to update the class means in nearest-mean classification.

Let p_{t-1}^c be the prototype of the class c at task $t-1$. Our aim is to update p_{t-1}^c after the training of task t to account for backbone drift. These updated prototypes are used for proto-

type rehearsal for training the model classifier. As proposed by SDC (Yu et al., 2020), we update the prototypes after the backbone is trained for each task as follows:

$$p_{t-1}^c \leftarrow p_{t-1}^c + \frac{\sum_{x_i \sim \mathcal{X}_t} w_i \delta_i}{\sum_{x_i \sim \mathcal{X}_t} w_i}, \quad \delta_i = f(x_i; \theta_t) - f(x_i, \theta_{t-1}), \quad (15)$$

where δ_i is the feature drift of samples x_i between task $t-1$ and t .

We define the weights w_i using the EFM (Figure 2b). In Section 3.3 we showed that the EFM induces a pseudo-metric in feature space, providing a second-order approximation of the KL-divergence due to feature drift. We extend the idea of feature drift estimation to softmax-based classifiers by weighting the overall drift of our class prototypes. Writing $p^c = p_{t-1}^c$ and $f_{t-1}(x_i) = f(x_i; \theta_{t-1})$ to simplify notation, our weights are defined as:

$$w_i = \exp\left(-\frac{(f_{t-1}(x_i) - p^c)E_{t-1}(f_{t-1}(x_i) - p^c)^\top}{2\sigma^2}\right) \\ \approx \exp\left(-\frac{\text{KL}(p(y|f_{t-1}(x_i); W_{t-1}) || p(y|p^c; W_{t-1}))}{2\sigma^2}\right) \quad (16)$$

where E_{t-1} is the EFM after training task $t-1$. Eq. 16 assigns higher weights to the samples more closely aligned to prototypes in terms of probability distribution. Specifically, higher weights are assigned to samples whose softmax prediction matches that of the prototypes, indicating a strong similarity for the classifier.

4.5 Inter-task Classifier Adaptation via Prototype Re-balancing

During the post-training stage, which we call *prototype re-balancing* phase (Figure 2c), we train all classifiers up to the current task using the prototypes updated using Eq. 15 and freezing the feature extractor:

$$\mathcal{L}_t^{\text{post-train}} := \mathcal{L}_t^{\text{ce}}(\tilde{\mathbf{P}} \cup f(\mathbf{X}, \theta_t^*); W_t), \quad (17)$$

where θ_t^* represents the frozen weights of the feature extractor and $W_t = [W_{t-1}, W_{t-1:t}]$ represents the learnable weights of the final classifier for the previous and current task classes. In this phase, we construct mini-batches by mixing prototypes, $\tilde{p}_{t-1}^c \in \tilde{\mathbf{P}}$, sampled from a Gaussian distribution $\mathcal{N}(p_{t-1}^c, \Sigma^c)$, with current task representations obtained from the frozen feature extractor $f(\mathbf{X}, \theta_t^*)$. The covariance matrix Σ^c represents the feature covariance of class c , computed in the task where class c was introduced. Prototypes and current task data are uniformly sampled from all encountered

classes to ensure balanced representation. Since the goal is to enable the final classifier to distinguish among classes from different tasks, the cross-entropy loss is evaluated over all encountered classes.

4.6 Elastic Feature Consolidation++

In summary, the training process begins with feature extractor training, where we optimize Eq. 14 to learn the new task while simultaneously regularizing relevant directions from previous tasks (Figure 2a). Once the feature extractor is trained, we update the prototypes using the EFM (see Eqs. 15, 16 in Section 4.4), ensuring that representations from the previous task remain aligned with the current task drift (Figure 2b).

After updating the prototypes, we perform the *prototype re-balancing* phase to train the classifier (Eq. 17). Since the prototypes have already been aligned with the updated feature extractor, this guarantees that the classifier is trained with consistent representations (Figure 2c). Finally, new prototypes and their corresponding class covariances are computed. The Empirical Feature Matrix E_t is then calculated using the current task data and is used as regularizer for the next incremental task (Figure 2d).

We call this procedure, detailed in Algorithm 1, Elastic Feature Consolidation++ (EFC++) to highlight its extension of our previous approach, Elastic Feature Consolidation (EFC), introduced in Magistri et al. (2024b).

Addressing PR-ACE limitations. EFC++ addresses the limitations of PR-ACE by decoupling the feature extractor training (Eq. 14) from inter-task classifier adaptation, which is instead performed after feature learning through prototype re-balancing (Eq. 17). This approach ensures that no additional drift occurs in the feature extractor while learning the current task, unlike in PR-ACE. Moreover, because EFC++ updates the prototypes before the post-training phase, the final classifier is trained using prototypes that more accurately reflect the real old class representations than those used by PR-ACE in EFC. As a result, *EFC++ more effectively controls drift in directions relevant to previous task classifiers without compromising plasticity* (see Figure 4, right).

To validate this claim, we consider a model continuously trained using either EFC or EFC++ up to task $t - 1$. We then measure the drift of the class means before and after training task t . Specifically, for each class $c \in \mathcal{C}_t$, we denote the feature representation of class c prior to task t by μ_{t-1}^c and after training task t by μ_t^c . These representations are obtained by averaging the feature representations of the data for class c . We define the feature drift of the class mean as the difference between these feature representations, specifically,

$$\Delta\mu^c = \mu_t^c - \mu_{t-1}^c. \quad (18)$$

Algorithm 1: EFC++: Elastic Feature Consolidation with Prototype Re-balancing

Data: $\mathcal{M}_1, E_1, P_1$

for $t = 2, \dots, T$ **do**

Initialize $\theta_t \leftarrow \theta_{t-1}$ and extend the classifier

$W_t = [W_{t-1}, W_{t-1:t}];$

/* Backbone Training */

for each optimization step do

sample $\mathbf{X} \sim \mathcal{X}_t$;

compute $\mathcal{L}_t^{\text{EFM}}$ (Eq. 10);

$\mathcal{L}_t^{\text{train}} = \mathcal{L}_t^{\text{EFM}} + \mathcal{L}_t^{\text{ce}}(f(\mathbf{X}, \theta_t); W_{t-1:t})$ (Eq. 14);

Update $\theta_t \leftarrow \theta_t - \alpha_1 \frac{\partial \mathcal{L}_t^{\text{train}}}{\partial \theta_t}$;

Update $W_{t-1:t} \leftarrow W_{t-1:t} - \alpha_2 \frac{\partial \mathcal{L}_t^{\text{train}}}{\partial W_{t-1:t}}$;

end

Set $\theta_t^* \leftarrow \theta_t$ /* freeze the feature extractor */

/* Prototype Update */

Update $P_{1:t-1}$ using Eqs. 15, 16;

/* Prototype Re-balancing */

for each optimization step do

sample $\mathbf{X} \sim \mathcal{X}_t, \tilde{\mathbf{P}} \sim P_{1:t-1}$;

$\mathcal{L}_t^{\text{post-train}} = \mathcal{L}_t^{\text{ce}}(\tilde{\mathbf{P}} \cup f(\mathbf{X}, \theta_t^*); W_t)$ (Eq. 17)

Update $W_t \leftarrow W_t - \alpha_3 \frac{\partial \mathcal{L}_t^{\text{post-train}}}{\partial W_t}$

end

Compute P_t from current-task features;

$P_{1:t} \leftarrow P_{1:t-1} \cup P_t$;

$E_t \leftarrow \text{EFM}(\mathcal{X}_t, \mathcal{M}_t)$ (Eq. 8) /* Compute EFM */

;

end

The squared pseudo-norm of $\Delta\mu^c$ via E_t , denoted as

$$\|\Delta\mu^c\|_{E_t}^2 = (\Delta\mu^c)^\top E_t \Delta\mu^c, \quad (19)$$

measures the magnitude of the drift of the class mean along the relevant directions identified after the training of task t . We then define the drift of task t as the average of the drifts of the class means, expressed as

$$\Delta = \frac{1}{|\mathcal{C}_t|} \sum_{c \in \mathcal{C}_t} \|\Delta\mu^c\|_{E_t}^2. \quad (20)$$

Figure 5 illustrates that, with EFC, the drift in the relevant directions due to PR-ACE is more pronounced in the earlier tasks compared to EFC++. In EFC++, decoupling the feature extractor training phase from the inter-task classifier learning phase allows for better control of the drift in the relevant directions. The reduction in drift along these directions does not compromise learning on the new classes. This is evidenced in Figure 4, which shows that the values on the diagonal of the confusion matrix for both EFC and EFC++ are comparable.

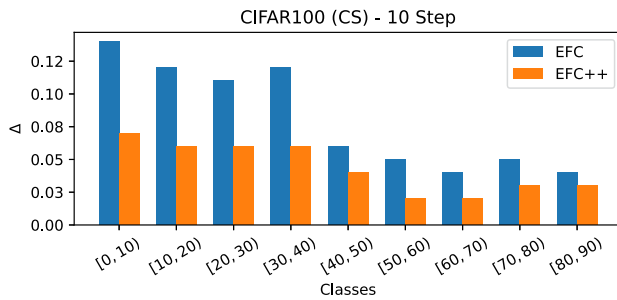


Fig. 5 Average drift of the class means in the relevant directions of the EFM before and after training the task in which they are involved, in both EFC and EFC++ on CIFAR-100 (CS) 10-step. EFC++ consistently exhibits less drift than EFC, especially in the initial tasks, where the drift of the classes is more pronounced (double) for EFC. This confirms that EFC++ better controls the drift along relevant directions

5 Experimental Settings

In this section, we describe the experimental setup used to evaluate Elastic Feature Consolidation++ (EFC++) against the state-of-the-art in Exemplar-free Class Incremental Learning (EFCIL).

Datasets. We evaluate EFC++ on five distinct datasets, with their statistics detailed in Appendix D.1. We categorize CIFAR-100 (Krizhevsky and Hinton, 2009), Tiny-ImageNet (Wu et al., 2017), and ImageNet-Subset (Deng et al., 2009) as *small-scale* datasets, containing 100, 200, and 100 classes, respectively. For a more comprehensive *large-scale* scenario, we use ImageNet-1K (Deng et al., 2009), which includes 1000 diverse object classes with varying numbers of samples per class. Additionally, we utilize DN4IL, a well-crafted and balanced subset of the standard Domain-Net dataset (Gowda et al., 2023). DN4IL consists of 600 classes, with 100 classes per domain, specifically *clipart*, *infograph*, *painting*, *quickdraw*, *real*, and *sketch*. This dataset is instrumental in comparing our approach to the state-of-the-art under *large domain shifts*.

Warm and Cold Start. We consider two scenarios for evaluating all approaches. The first, which we refer to as the **Warm Start (WS)** scenario which is commonly considered in the EFCIL literature (Zhu et al., 2021b, 2022; Petit et al., 2023; Toldo and Ozay, 2022), involves pre-training the model on 50% or 40% of all available classes, denoted as $\mathcal{C}_0$. The remaining classes are then equally distributed among the subsequent incremental learning steps. This setting, which begins with a larger task favors continual learning techniques that prioritize *stability* in retained knowledge over the *plasticity* required to learn new classes.

The second scenario, which we call **Cold Start (CS)**, follows the setting in EFC (Magistri et al., 2024b). In this scenario, the model is randomly initialized before incremental learning, and all available classes are equally split among the incremental learning steps, resulting in no initial pre-

training phase. Consequently, the number of classes per step is larger compared to Warm Start. See Table 1 for a detailed description of how the classes are distributed in Warm Start and Cold Start across all datasets.

We are primarily interested in the Cold Start scenario since it demands high backbone plasticity and poses a significant challenge for EFCIL. Mitigating forgetting becomes difficult without using exemplars when larger backbone shifts occur (in Section 7.2 we provide an empirical analysis of this phenomenon).

Metrics. We use the **per-step incremental accuracy** A_{step}^K and **average incremental accuracy** A_{inc}^K :

$$A_{\text{step}}^K = \frac{\sum_{i=s}^K |\mathcal{C}_i| a_i^K}{\sum_{i=s}^K |\mathcal{C}_i|}, \quad A_{\text{inc}}^K = \frac{1}{K-s+1} \sum_{i=s}^K A_{\text{step}}^i, \quad (21)$$

where a_i^K represents the accuracy on task i after training task K , and $s = 1$ in the Cold Start scenario and $s = 0$ in the Warm Start scenario. It should be noted that in the Warm Start (WS) scenario, the performance on the initial classes, indicated as step $s = 0$, is commonly included in the evaluation metric (Zhu et al., 2021b; Toldo and Ozay, 2022; Zhu et al., 2022; Petit et al., 2023; Goswami et al., 2023), and we follow this same approach. Therefore, achieving higher stability on the large number of initial classes significantly impacts the final evaluation. In contrast, in the Cold Start (CS) scenario the classes are equally distributed among the incremental learning tasks and the evaluation metrics starting from $s = 1$ give equal importance to each step.

To gain a more comprehensive understanding of impact of different regularization techniques employed by the tested methods, we use **average forgetting** F^K to measure stability during training and **cumulative average accuracy on the last task** PL^K to measure plasticity (i.e. the average of the diagonal of the matrix in Figure 4) (Lange et al., 2023). More specifically, the forgetting of a task is the difference between its maximum accuracy obtained in the past and its current accuracy:

$$f_j^K = \max_{i \in \{s, \dots, k-1\}} (a_j^i - a_j^K), \quad j < K. \quad (22)$$

Thus, F^K and PL^K at the K -th task are defined as:

$$F^K = \frac{1}{K-s} \sum_{j=s}^{K-1} f_j^K, \quad \text{PL}^K = \frac{1}{K-s+1} \sum_{i=s}^K a_i^K. \quad (23)$$

Hyperparameter settings. We use standard ResNet-18 (He et al., 2015) trained from scratch for all experiments. We train the first task of each state-of-the-art method using the same optimizer, number of epochs, and data augmentations. In particular, we train the first task using self-rotation as in Zhu et

Table 1 Class Distribution in Warm Start and Cold Start: $|C_0|$ is the number of classes in the pre-training phase, while $|C_{i>0}|$ represents classes introduced per incremental step

Dataset	Warm Start (WS)				Cold Start (CS)			
	10 Step		20 Step		10 Step		20 Step	
	$ C_0 $	$ C_{i>0} $	$ C_0 $	$ C_{i>0} $	$ C_0 $	$ C_{i>0} $	$ C_0 $	$ C_{i>0} $
CIFAR-100	50	5	40	3	0	10	0	5
Tiny-ImageNet	100	10	100	5	0	20	0	10
ImageNet-Subset	50	5	40	3	0	10	0	5
ImageNet-1K	500	50	400	30	0	100	0	50

al. (2021b); Toldo and Ozay (2022) to align the performance of all techniques.

For the incremental steps of EFC++ we used Adam with a weight decay of $2e-4$ and a fixed learning rate of $1e-4$ for Tiny-ImageNet and CIFAR-100, while for ImageNet-Subset, ImageNet-1K, DN4IL we use a learning rate of $1e-5$ for the backbone and $1e-4$ for the heads. We fixed the total number of epochs to 100 and use a batch size of 64, except for ImageNet-1K where the batch size is fixed to 256. We set $\lambda_{\text{EFM}} = 10$ and $\eta = 0.1$ in Eq. 10 for all the experiments. In Section 7.3, we discuss the theoretical basis for selecting these hyperparameters and analyze their impact on performance. For EFC++, during post-training we use SGD with a fixed learning rate $1e-3$ to train the final classifier on both prototype and current task features for 50 epochs.

We ran all approaches using five random seeds and shuffling the classes in order to reduce the bias induced by the choice of class ordering (Rebuffi et al., 2017; Masana et al., 2022). In Appendix D.2 we provide the optimization settings for each state-of-the-art method we evaluated.

6 Experimental Results

In this section we compare EFC++ with state-of-the-art Exemplar-free Class-Incremental Learning (EFCIL) methods across small-scale Class-Incremental (Class-IL) benchmarks (CIFAR-100, Tiny-ImageNet, and ImageNet Subset), large-scale Class-IL ImageNet-1K, and the domain- and class-incremental DN4IL benchmark.

We organize the discussion around two main questions: whether EFC++ remains competitive in the standard Warm Start setting despite the increased plasticity induced by the EFM, and whether its advantage increases in the more challenging Cold Start setting. We mainly discuss the last per-step accuracy A_{step}^K , while the average incremental accuracy A_{inc}^K follows the same trends.

6.1 Small-scale Class-IL Experiments

Table 2 compares EFC++ with baseline and recent state-of-the-art EFCIL approaches, including EWC (Kirkpatrick et

al., 2017), LwF (Li and Hoiem, 2017), PASS (Zhu et al., 2021b), Fusion (Toldo and Ozay, 2022), FeTrIL (Petit et al., 2023), SSRE (Zhu et al., 2022), FeCAM (Goswami et al., 2023), ABD (Smith et al., 2021), R-DFCIL (Gao et al., 2022), and EFC (Magistri et al., 2024b). The comparison covers both Warm Start (WS) and Cold Start (CS), using protocols with 10 and 20 incremental learning steps. All methods are evaluated under the same first-task training and data augmentation protocol. This may lead to differences from originally reported SSRE and FeCAM results, especially in WS where the large first task strongly affects the final metrics (Eq. 21)(Zhou et al., 2023b; Castro et al., 2018). Further implementation details and per-step plots are reported in Appendices D.1, J.1, and J.2.

Warm Start. EFC++ consistently improves over methods based on feature distillation and prototype rehearsal, such as PASS, Fusion, and SSRE, as well as over FeTrIL, which freezes the feature extractor, with gains typically ranging from 2 to 6 percentage points. Methods based on generative inversion, such as ABD and R-DFCIL, perform worse than both feature distillation methods and methods that freeze the backbone, suggesting that inversion is less effective when the large first task dominates the incremental protocol. Compared to EFC, EFC++ achieves consistent but moderate improvements. For CIFAR-100 and Tiny-ImageNet, the gains over EFC are approximately 1.3% and 1.7% in the 10-step and 20-step settings, respectively, while they are smaller on ImageNet-Subset. This indicates that prototype re-balancing is still beneficial in WS, although its impact is reduced compared to Cold Start because the stronger initial backbone limits feature drift and, consequently, the misalignment between prototypes and features induced by PR-ACE in EFC. This trend is further analyzed in Section 7.2.

Finally, EFC++ remains competitive with FeCAM, the strongest competitor in WS. The two methods are comparable on CIFAR-100, while on Tiny-ImageNet and ImageNet-Subset, EFC++ improves performance by about 1.2% in the 10-step setting and remains comparable in the 20-step setting. This is noteworthy because FeCAM freezes the backbone after the large first task, which preserves first task accuracy and strongly influences the per-step accuracy in WS. In contrast, EFC++ keeps updating the feature extractor through

Table 2 Small-scale Class-IL experiments. Fusion results are those reported in Toldo and Ozay (2022) as no code is provided to reproduce them. The result for ImageNet-Subset Cold Start for ABD and R-DFCIL are those reported in Gao et al. (2022)

	Method	Warm Start (WS)				Cold Start (CS)			
		A_{step}^K		A_{inc}^K		A_{step}^K		A_{inc}^K	
		10 Step	20 Step	10 Step	20 Step	10 Step	20 Step	10 Step	20 Step
CIFAR-100	EWC	21.08 ± 1.09	13.53 ± 1.11	41.00 ± 1.11	31.79 ± 2.77	31.17 ± 2.94	17.37 ± 2.43	49.14 ± 1.28	31.02 ± 1.15
	LwF	20.73 ± 1.47	11.78 ± 0.60	41.95 ± 1.30	28.93 ± 1.62	32.80 ± 3.08	17.44 ± 0.73	53.91 ± 1.67	38.39 ± 1.05
	PASS	53.42 ± 0.48	47.51 ± 0.37	63.42 ± 0.69	59.55 ± 0.97	30.45 ± 1.01	17.44 ± 0.69	47.86 ± 1.93	32.86 ± 1.03
	Fusion	56.86	51.75	65.10	61.60	—	—	—	—
	FeTrL	56.79 ± 0.40	52.61 ± 0.81	65.03 ± 0.66	62.50 ± 1.03	34.94 ± 0.46	23.28 ± 1.24	51.20 ± 1.13	38.48 ± 1.07
	SSRE	57.48 ± 0.55	52.98 ± 0.63	65.78 ± 0.59	63.11 ± 0.84	30.40 ± 0.74	17.52 ± 0.80	47.26 ± 1.91	32.45 ± 1.07
	FeCAM	62.13 ± 0.51	58.74 ± 0.60	68.87 ± 0.40	67.42 ± 0.88	37.63 ± 0.52	23.82 ± 0.87	52.53 ± 1.19	38.51 ± 0.82
	ABD [†]	44.91 ± 1.00	33.22 ± 1.11	56.81 ± 1.45	47.55 ± 0.19	43.11 ± 0.60	28.93 ± 0.21	59.14 ± 1.72	47.12 ± 0.85
	R-DFCIL [†]	52.24 ± 0.32	40.97 ± 0.12	63.09 ± 0.21	55.65 ± 0.10	42.14 ± 0.54	30.35 ± 0.05	57.77 ± 0.50	47.41 ± 0.27
	EFC	60.87 ± 0.39	55.78 ± 0.42	68.23 ± 0.68	65.90 ± 0.97	43.62 ± 0.70	32.15 ± 1.33	58.58 ± 0.91	47.36 ± 1.37
Tiny-ImageNet	EFC++	62.15 ± 0.48	57.55 ± 0.66	69.26 ± 0.63	67.11 ± 0.91	47.52 ± 0.68	33.71 ± 1.41	61.57 ± 0.66	49.20 ± 1.10
	EWC	6.73 ± 0.44	5.96 ± 1.17	18.48 ± 0.60	13.74 ± 0.52	8.00 ± 0.27	5.16 ± 0.54	24.01 ± 0.51	15.70 ± 0.35
	LwF	24.00 ± 1.44	7.58 ± 0.37	43.15 ± 1.02	22.89 ± 0.64	26.09 ± 1.29	15.02 ± 0.67	45.14 ± 0.88	32.94 ± 0.54
	PASS	41.67 ± 0.64	35.01 ± 0.39	51.18 ± 0.31	46.65 ± 0.47	24.11 ± 0.48	18.73 ± 1.43	39.25 ± 0.90	32.01 ± 1.68
	Fusion	46.92	44.61	—	—	—	—	—	—
	FeTrL	45.71 ± 0.39	44.63 ± 0.49	53.95 ± 0.42	52.96 ± 0.45	30.97 ± 0.90	25.70 ± 0.61	45.60 ± 1.67	39.54 ± 1.19
	SSRE	44.66 ± 0.45	44.68 ± 0.36	53.27 ± 0.43	52.94 ± 0.42	22.93 ± 0.95	17.34 ± 1.06	38.82 ± 1.99	30.62 ± 1.96
	FeCAM	50.77 ± 0.62	50.73 ± 0.46	57.67 ± 0.48	57.57 ± 0.37	30.79 ± 0.69	25.36 ± 0.74	45.00 ± 1.35	38.59 ± 1.29
	ABD [†]	24.07 ± 1.08	11.17 ± 1.30	44.92 ± 1.41	34.70 ± 1.38	23.24 ± 0.57	12.21 ± 0.65	42.80 ± 1.07	33.37 ± 1.20
	R-DFCIL [†]	43.06 ± 0.12	36.55 ± 0.06	53.39 ± 0.20	49.89 ± 0.25	35.49 ± 0.25	28.23 ± 0.17	49.14 ± 0.35	42.30 ± 0.17
	EFC	50.40 ± 0.25	48.68 ± 0.65	57.52 ± 0.43	56.52 ± 0.53	34.10 ± 0.77	28.69 ± 0.40	47.95 ± 0.61	42.07 ± 0.96
	EFC++	51.67 ± 0.31	50.41 ± 0.51	58.54 ± 0.51	57.79 ± 0.54	37.48 ± 0.52	32.56 ± 0.44	50.18 ± 1.11	45.58 ± 1.32

Table 2 continued

	Warm Start (WS)				Cold Start (CS)			
	Method	A_{step}^K		A_{inc}^K	A_{step}^K		A_{inc}^K	
		10 Step	20 Step		10 Step	20 Step		
ImageNet-Subset	EWC	16.19 ± 2.48	10.66 ± 1.74	23.58 ± 2.01	18.05 ± 1.10	24.59 ± 4.13	12.78 ± 1.95	26.95 ± 1.02
	LwF	21.89 ± 0.52	13.24 ± 1.61	37.15 ± 2.47	25.96 ± 0.95	37.71 ± 2.53	18.64 ± 1.67	40.23 ± 0.43
	PASS	52.04 ± 1.06	44.03 ± 2.19	65.14 ± 0.36	58.88 ± 2.15	26.40 ± 1.33	14.38 ± 1.22	31.65 ± 0.42
	Fusion	60.20	51.60	70.00	63.70	—	—	—
	FeTrIL	63.56 ± 0.59	57.62 ± 1.13	71.87 ± 1.46	68.01 ± 1.60	36.17 ± 1.18	26.63 ± 1.45	42.43 ± 2.05
	SSRE	61.84 ± 0.93	55.19 ± 0.97	70.68 ± 1.37	66.73 ± 1.61	25.42 ± 1.17	16.25 ± 1.05	31.15 ± 1.53
	FeCAM	67.68 ± 0.45	62.74 ± 1.04	74.59 ± 0.94	71.72 ± 1.45	35.54 ± 1.20	26.03 ± 1.33	40.87 ± 2.00
	ABD	38.68 ± 0.76	21.47 ± 2.08	55.72 ± 1.06	40.92 ± 2.30	35.96	22.40	44.89
	R-DFCIL	49.81 ± 0.23	38.19 ± 0.13	64.50 ± 0.31	56.8 ± 0.69	42.28	30.28	47.33
	EFC	68.85 ± 0.58	62.17 ± 0.69	75.40 ± 0.92	71.63 ± 1.13	47.38 ± 1.43	35.75 ± 1.74	49.92 ± 2.05
	EFC++	69.28 ± 0.46	62.75 ± 0.46	75.71 ± 1.00	72.05 ± 1.33	53.90 ± 1.18	40.86 ± 1.65	55.42 ± 2.44

[†] ABD and R-DFCIL are run on CIFAR-100 and Tiny-ImageNet with ResNet-18 for consistency, although the original papers used ResNet-32 except on ImageNet-Subset

EFM regularization, improving plasticity for later tasks while maintaining competitive stability on the initial task. The WS panel of Figure 6 confirms this behavior: FeCAM better preserves the large initial task, while EFC++ obtains stronger accuracy on the other tasks after the last training step.

Cold Start. In Cold Start (Table 2, right), methods based on feature distillation, such as SSRE and PASS, and methods that freeze the feature extractor, such as FeTrIL and FeCAM, show a substantial drop in accuracy compared to EFC++. This confirms that, when no strong initial representation is available, excessive stability limits the ability to learn later tasks. Most of these methods also perform worse than LwF, despite using prototypes for classifier adaptation, suggesting that prototypes alone are not sufficient when the backbone lacks plasticity.

Interestingly, R-DFCIL reverses its behavior with respect to Warm Start: despite its weak WS performance, it becomes the strongest non-EFC baseline in CS. This suggests that its knowledge distillation loss increases plasticity, improving the learning of new tasks and therefore the final per-step accuracy, where all tasks contribute equally to the metric in CS (see Eq. 21). Nevertheless, EFC++ outperforms R-DFCIL across all benchmarks in both incremental protocols: by 5% and 3% on CIFAR-100, 2% and 4% on Tiny-ImageNet, and 11% and 10% on ImageNet-Subset for the 10-step and 20-step settings, respectively. The larger gap on ImageNet-Subset suggests that the generative mechanism used by R-DFCIL to rehearse old class samples becomes more challenging as image resolution increases (224×224 for ImageNet-Subset). The CS panel of Figure 6 confirms this trend: after the final incremental step, EFC++ achieves higher accuracy than R-DFCIL across all tasks, while FeCAM achieves better performance only on the first task.

Compared with EFC, EFC++ achieves consistent and substantial gains of approximately 3–4 percentage points across all CS settings. This improvement is due to the decoupling of backbone training from classifier head balancing. In EFC, prototypes are used while the feature space is still evolving, so they can become outdated during training. EFC++ avoids this issue by using the updated prototypes only after backbone training, which is especially important in Cold Start, where feature drift is significant (see Section 7.2 for the drift comparison in WS and CS).

6.2 Large-scale Domain- and Class-IL Experiments

We next evaluate EFC++ on ImageNet-1K to assess its scalability to a large-scale class-incremental setting, and on DN4IL (a subset of DomainNet) to test its performance in a more challenging scenario involving both class-incremental and domain-incremental shifts. Here, we report the final per-step accuracy in CS; the average incremental accuracy in CS

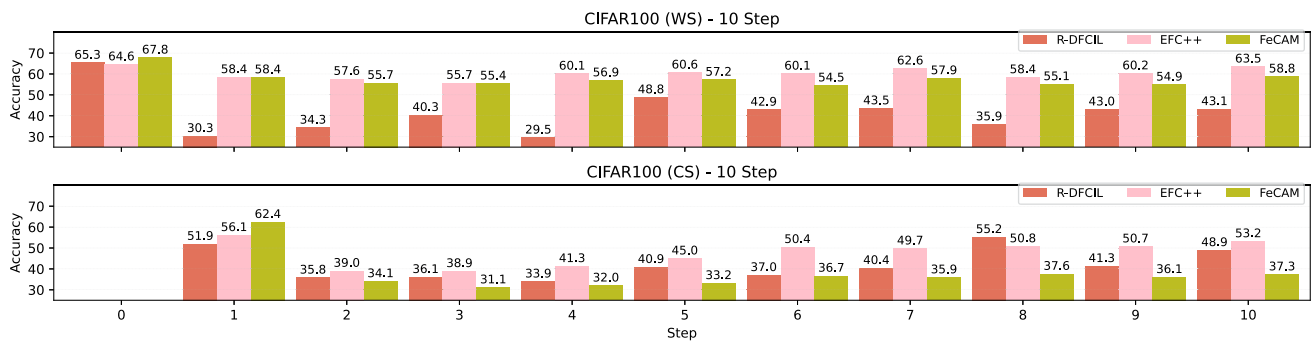


Fig. 6 Accuracy on each task after the final training step on CIFAR-100 Warm Start and Cold Start. EFC++ achieves the best stability-plasticity trade-off compared to the nearest competitor in Warm Start (FeCAM)

and Cold Start (R-DFCIL). Task 0 represents the large first task in WS, which is absent in CS (see Table 1)

Table 3 Large scale Class- and Domain-IL Experiments on ImageNet-1K (CS) and DN4IL (Gowda et al., 2023) (CS) using A_{step}^K

	ImageNet-1K (CS)		DN4IL (CS)
Method	10 Step	20 Step	6 Step
FeTrIL	34.28 ± 0.20	26.64 ± 0.25	30.95 ± 0.17
FeCAM	36.16 ± 0.10	27.24 ± 0.19	36.42 ± 0.12
R-DFCIL	29.36 ± 0.17	22.30 ± 0.19	22.31 ± 0.21
EFC	42.62 ± 0.08	36.32 ± 0.35	38.27 ± 0.16
EFC++	44.72 ± 0.11	37.46 ± 0.03	39.32 ± 0.15

and WS experiments are provided in Appendix F, and follow the same trends observed on ImageNet-Subset.

Experiments on ImageNet-1K. EFC++ achieves the best per-step accuracy in both the 10-step and 20-step CS settings, improving over EFC by 2.1 and 1.2 percentage points, respectively. Freezing the feature extractor limits the plasticity of FeTrIL and FeCAM, leading to drops of more than 6 and 8 percentage points, respectively, compared to EFC and EFC++. Although R-DFCIL performs well in small-scale experiments, it underperforms FeTrIL and FeCAM in this setting, likely because its generative inversion network must synthesize high-resolution images for 1,000 distinct classes by the end of the training, ten times more than ImageNet-Subset.

Experiments on DN4IL (DomainNet). As shown in Table 3, EFC++ improves upon EFC by approximately 1.1% in final per-step accuracy and outperforms the remaining state-of-the-art approaches. Notably, EFC++ surpasses the best competitor, FeCAM, by roughly 3 points. R-DFCIL struggles here for reasons similar to those observed on ImageNet-Subset. Additional comparisons are provided in Appendix F. Figure 7 reports the per-task accuracy after the last training phase (domain in this setting). EFC++ remains competitive on the first domain compared to methods freezing the backbone, while showing better plasticity on later domains. The

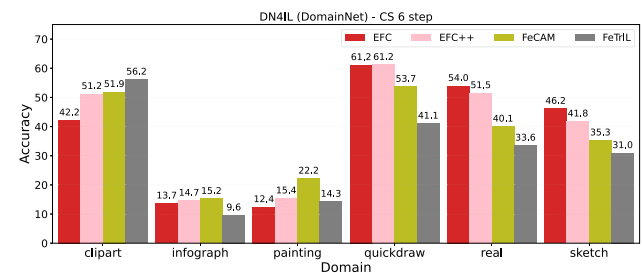


Fig. 7 Accuracy on each domain after the final training step on DN4IL. Methods that freeze the backbone, such as FeCAM and FeTrIL, achieve higher accuracy on the first domain, while EFC and EFC++ excel at learning new domains

main exception is the *painting* domain, where FeCAM performs better and FeTrIL is comparable, possibly because this domain is more strongly correlated with the first one. Compared with EFC, EFC++ achieves a better stability–plasticity trade-off, as EFC loses nearly 10 points on the first domain.

6.3 Comparison using the PyCIL framework with recent EFCIL methods

To compare with very recent exemplar-free Class-IL methods, we additionally evaluate EFC++ within the PyCIL framework (Zhou et al., 2023a, 2024, ?). We use the DPCR codebase (He et al., 2025) to ensure a fair comparison with recent methods, adopting the same data augmentations, first-task optimization protocol, class order, and random seed. We note that these results are *not* directly comparable with those in Table 2, since PyCIL uses different data augmentations and evaluates ImageNet-100, whose class subset differs from the ImageNet-Subset benchmark from PODNet (Douillard et al., 2020) used in our main experiments.

Table 4 compares EFC++ with DS-AL, ADC, LDC, and the recent DPCR method. On CIFAR-100, DPCR obtains the best results, while EFC++ remains competitive and ranks second in both the 10-step and 20-step settings. On Tiny-

Table 4 Final per-step accuracy A_{step}^K on CIFAR-100, Tiny-ImageNet, and ImageNet-100 in Class-IL Cold Start. Results are obtained using the DPCR codebase (He et al., 2025), based on the PyCIL framework (Zhou et al., 2023a)

Dataset	Method	10 Step	20 Step
CIFAR-100	DS-AL (2024)	36.83	28.90
	ADC (2024)	46.80	34.69
	LDC (2025)	46.60	36.76
	DPCR (2025)	50.24	38.98
	EFC++	<u>49.47</u>	<u>36.85</u>
Tiny-ImageNet	DS-AL (2024)	27.01	21.86
	ADC (2024)	32.90	20.69
	LDC (2025)	33.74	24.49
	DPCR (2025)	<u>35.20</u>	<u>26.54</u>
	EFC++	38.66	32.65
ImageNet-100	DS-AL (2024)	45.55	34.10
	ADC (2024)	46.69	32.21
	LDC (2025)	49.98	34.87
	DPCR (2025)	<u>52.16</u>	<u>38.35</u>
	EFC++	55.86	38.60

ImageNet, EFC++ improves over DPCR by 3.46 and 6.11 points in the 10-step and 20-step settings, respectively. On ImageNet-100, EFC++ obtains the best results, improving over DPCR by 3.70 and 0.25 points. These results show that EFC++ remains competitive with the latest methods on CIFAR-100 and achieves stronger performance on Tiny-ImageNet and ImageNet-100 under the same PyCIL protocol.

A possible explanation for the comparatively smaller gains on CIFAR-100 is that its lower image resolution and relatively limited scale may make learning transferable feature representations from scratch in the Cold Start setting more difficult. Despite the strong data augmentations used in PyCIL, we observed signs of overfitting during training, consistent with the findings of Mittal et al. (2021), who showed that overfitting during first-task training degrades representation quality and class-incremental learning performance on CIFAR-100. Consequently, EFM-based feature consolidation has less opportunity to exploit robust feature representations, leading to smaller improvements than on larger, more diverse datasets.

7 Ablations and Additional analysis

In this section, we systematically ablate the key components of EFC++ to assess their individual contributions. We then analyze differences in feature representation drift between Warm Start and Cold Start scenarios, followed by a spectral

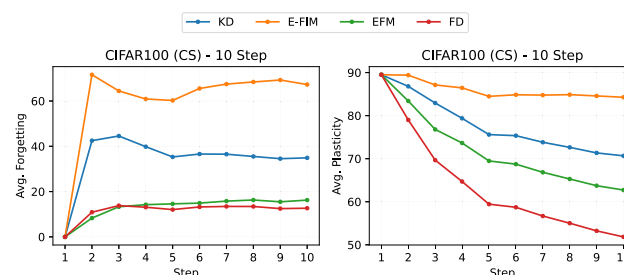


Fig. 8 Comparison of regularization methods on CIFAR-100 (CS) 10-step. EFM balances stability and plasticity, reducing forgetting (left) and improving plasticity (right) over FD. KD and EWC exhibit strong initial plasticity but suffer from severe forgetting after the first task

analysis of the Empirical Feature Matrix and the impact of hyperparameter selection for EFM regularization. Finally, we discuss computational and storage requirements and examine the main limitations of EFC++, focusing on prototype rehearsal and covariance drift.

7.1 Ablation Studies

Ablation on EFM regularizer. We evaluate our proposed regularizer, EFM, against three established regularizer: Fisher Information Matrix (E-FIM, Eq. 3), Feature Distillation (FD, Eq. 5), and Knowledge Distillation (KD) (Li and Hoiem, 2017). To carry out this comparison, we directly replaced EFM regularization in our training framework without modifying any other components. Specifically, we maintained the same prototype updates and prototype rebalancing procedures. In order to perform these experiments, we carefully validate the hyperparameters associated with each regularizer (see Appendix H for more details on the validation). For these experiments, in addition to the per-step accuracy, we report the average forgetting and the average plasticity (see Eq. 23), to measure the stability and the plasticity induced by each regularizer. As shown in Table 5, EFM achieves the best trade-off between stability and plasticity, leading to the highest final accuracy in both the 10-step and 20-step settings on CIFAR-100. Figure 8 further highlights EFM effectiveness: its average forgetting (left) remains comparable to FD during the incremental steps, while its plasticity (right) is significantly improved. In contrast, while KD and E-FIM exhibit strong plasticity (right), they suffer from severe forgetting (left) after the first task, ultimately degrading their final performance as shown in Table 5.

Ablation on prototype update. In Table 6 we evaluate the effect of prototype updates using Eq. 15 and Eq. 16. Prototype updates significantly improve performance in both Warm Start and Cold Start across most benchmarks, except for ImageNet-Subset WS, where no significant improvement is observed.

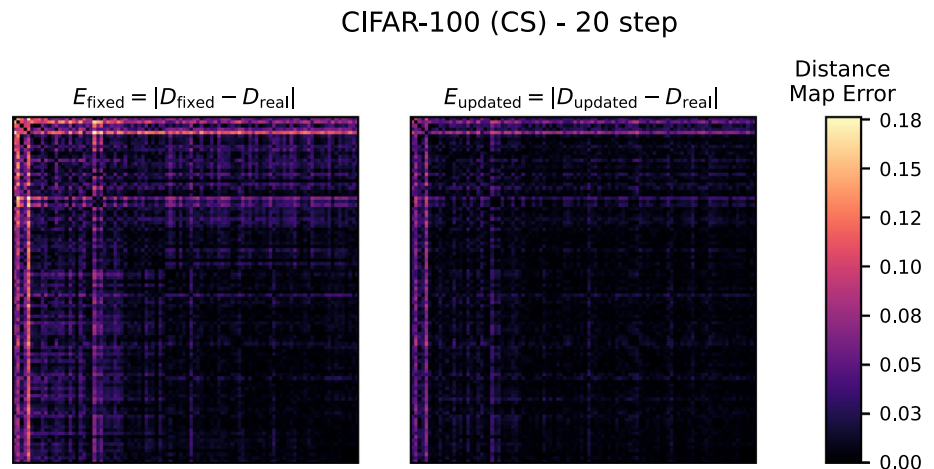
Table 5 Ablation study on regularizers. We report the final average forgetting (F^K), average plasticity (PL^K), and per-step accuracy (A_{step}^K) for CIFAR-100 at 10 and 20 steps

	Regularizer	CS - 10 Step			CS - 20 Step		
		$F^K(\downarrow)$	$PL^K(\uparrow)$	$A_{\text{step}}^K(\uparrow)$	$F^K(\downarrow)$	$PL^K(\uparrow)$	$A_{\text{step}}^K(\uparrow)$
CIFAR-100	E-FIM	67.28	84.27	23.72	71.98	83.22	14.83
	KD	34.90	<u>70.67</u>	39.26	41.11	<u>65.41</u>	26.39
	FD	12.67	51.83	<u>40.44</u>	8.78	35.85	<u>29.02</u>
	EFM	<u>16.24</u>	62.72	47.52	<u>18.52</u>	50.79	33.71

Table 6 Ablation study on prototype update. The performance of EFC++ improves with prototype updates in both Warm and Cold Start

Dataset	Step	Update Proto (WS)		Update Proto (CS)	
		$\times$	$\checkmark$	$\times$	$\checkmark$
CIFAR-100	10	60.26 $\pm$ 0.72	62.15 $\pm$ 0.48	46.30 $\pm$ 1.45	47.52 $\pm$ 0.68
	20	56.10 $\pm$ 0.34	57.55 $\pm$ 0.66	31.71 $\pm$ 2.26	33.71 $\pm$ 1.41
Tiny-ImageNet	10	50.57 $\pm$ 0.40	51.67 $\pm$ 0.31	36.17 $\pm$ 0.41	37.48 $\pm$ 0.52
	20	46.86 $\pm$ 1.19	50.41 $\pm$ 0.51	29.04 $\pm$ 0.65	32.56 $\pm$ 0.44
ImageNet-Subset	10	69.09 $\pm$ 0.30	69.28 $\pm$ 0.46	52.95 $\pm$ 1.00	53.90 $\pm$ 1.18
	20	62.77 $\pm$ 0.63	62.75 $\pm$ 0.46	39.10 $\pm$ 1.67	40.86 $\pm$ 1.65

Fig. 9 Inter-class distance-map analysis on CIFAR-100 Cold Start with 20 steps. We compare the absolute error between the cosine-distance maps computed from fixed prototypes (D_{fixed}) or EFM-updated prototypes (D_{updated}) and the map computed from real final class prototypes (D_{real}). EFM-updated prototypes better match the final inter-class geometry and reduce the average off-diagonal error, showing that the proposed update better preserves the feature-space structure across tasks



7.2 Analysis of Feature Drift in Warm and Cold Start

We further analyze how initial representation quality affects feature drift during incremental learning. Methods that freeze the backbone perform well in Warm Start but struggle in Cold Start, suggesting that strong initial features reduce the need for later plasticity. Consistently, EFC++ improves over EFC more clearly in Cold Start, where the induced drift of PR-ACE has a larger impact on final performance. This suggests that *representations are less prone to change in Warm Start thanks to the stronger initialization of the feature extractor.*

To empirically evaluate this, we propose applying EFC++ in two settings. In a Warm Start setting, the model is pre-trained on half of the available classes (i.e., $|C_0| = 50$ for CIFAR-100) and then incrementally trained for 10 steps, each with 5 classes per step ($|C_{i>0}| = 5$). In a Cold Start setting, the model starts with a random initialization (i.e., $|C_0| = 0$)

and is then incrementally trained for 10 steps by following the same order and the same number of classes per step as in the Warm Start setting. Figure 10 presents the results of this analysis. For each set of classes (on the x -axis), we plot two lines representing the average distance between the actual class means (μ) and the estimated prototypes (p) throughout the experiment, with and without updates (dashed and solid lines, respectively) using the Empirical Feature Matrix as pseudo-metric. Comparing the two scenarios, we see that the average distance between the real class means and fixed prototypes is greater in the Cold Start setting, indicating that the representations are more prone to change compared to the Warm Start setting. Additionally, our prototype update rule (see Equations (15) and (16)) effectively mitigates prototype drift in both scenarios. However, it is less effective in the more challenging Cold Start scenario due to the greater representation drift.

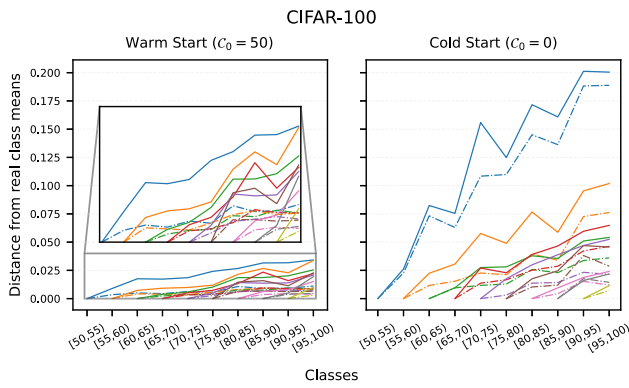


Fig. 10 Distance between real class means and prototypes when kept fixed (solid) or updated via the Empirical Feature Matrix (dashed) in both Warm and Cold Start scenarios on CIFAR-100 in EFC++. In Cold Start, the average distance of the prototypes from real class means change more than in Warm Start. Our prototype update rule effectively mitigates this drift

Visual analysis of inter-class feature geometry. To further assess whether the EFM-based prototype update preserves the structure of the feature space, we analyze inter-class distance maps on CIFAR-100 in the Cold Start 20-step setting. Let D_{fixed} , D_{updated} , and D_{real} denote the pairwise cosine-distance matrices computed from fixed prototypes stored when each class is first encountered, EFM-updated prototypes after the final training step, and real final class prototypes, respectively. The real final prototypes are obtained by forwarding the corresponding class data through the final backbone. We compare each prototype-based distance map with the real final one by computing $E_{\text{fixed}} = |D_{\text{fixed}} - D_{\text{real}}|$ and $E_{\text{updated}} = |D_{\text{updated}} - D_{\text{real}}|$.

Figure 9 illustrates E_{fixed} and E_{updated} . Fixed prototypes exhibit larger errors, indicating that the original inter-class geometry becomes misaligned as the feature extractor evolves. In contrast, EFM-updated prototypes produce a distance map closer to the real final one and reduce the average off-diagonal error. This confirms that the proposed update better preserves the relative geometry among classes across tasks.

7.3 EFM Spectral Analysis and Hyperparameters

Since EFC++ uses E_t to selectively regularize drift in feature space, it is natural to question *how* selective its regularization is. We can gain insight into this by analyzing how the rank of E_t evolves with increasing incremental tasks.

Consider training a model incrementally on CIFAR-100 using EFC++ up to the tenth and final task, computing along the way the spectrum of E_t for each $t \in [1, \dots, 10]$. We chose the ten task sequence for simplicity, but the same empirical conclusions hold regardless of the number of tasks. Recalling that $E_t \in \mathbb{R}^{n \times n}$ is symmetric, there then exist U_t

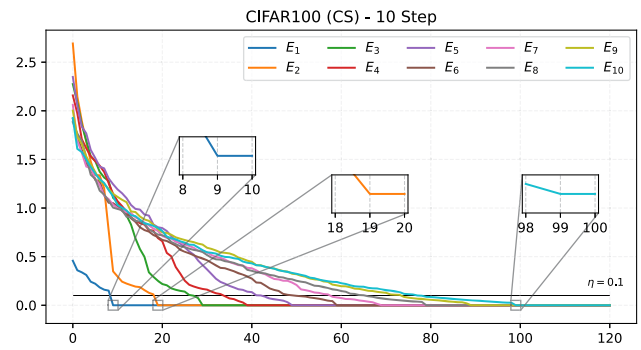


Fig. 11 The spectrum of the Empirical Feature Matrix across incremental learning steps. For a better visualization of the spectrum in the analysis we considered a Cold Start 10-step scenario on CIFAR-100. The x -axis is truncated at the 120th eigenvalue

and $\Lambda_t \in \mathbb{R}^{n \times n}$ such that $E_t = U_t^\top \Lambda_t U_t$, where U_t is an orthogonal matrix and $\Lambda_t = \text{diag}(\lambda_1, \dots, \lambda_n)$ is a diagonal matrix whose entries contain the eigenvalues of E_t , sorted in descending order. Since E_t is positive semi-definite, $\lambda_i \geq 0$ for each $i \in [1, \dots, n]$.

In Figure 11, we plot the spectrum of E_t at each $t \in [1, \dots, 10]$. This plot shows that the number of classes on which the matrix is estimated corresponds to an elbow in the curve beyond which the spectrum of the matrix vanishes. *This implies that the rank of the matrices is exactly equal to the number of observed classes minus one.*

Moreover, the horizontal line compares the magnitude of the eigenvalues across tasks with the damping term. Recalling that the results reported for EFC++ were obtained using parameter values of $\lambda_{\text{EFM}} = 10$ and $\eta = 0.1$, we observe that the plasticity constraints $\lambda_{\text{EFM}} v_i > \eta$ (described in Section 3.3) are satisfied for nearly every $v_i > 0$, clearly demonstrating that *our regularizer effectively constrains the features in a non-isotropic manner, distinguishing it from feature distillation.*

To provide empirical proof of the importance of calibrating such hyperparameters (i.e., avoiding too low a value of η or degenerating into feature distillation), we also conducted a grid search testing 25 different combinations of λ_{EFM} and η . Figure 12 (left) shows that the selected combination achieves the peak value in terms of final per-step accuracy. In particular, it reveals that excessively low values of η lead to weaker regularization and increased forgetting (middle), whereas excessively high values significantly reduce plasticity (right). See Appendix G for a more comprehensive analysis on other scenarios.

7.4 Training Time and Memory Requirements

Training Time. FeTrIL and FeCAM are the fastest approaches as they freeze the backbone. Next is EWC, which does not apply any distillation and only E-FIM regularization.

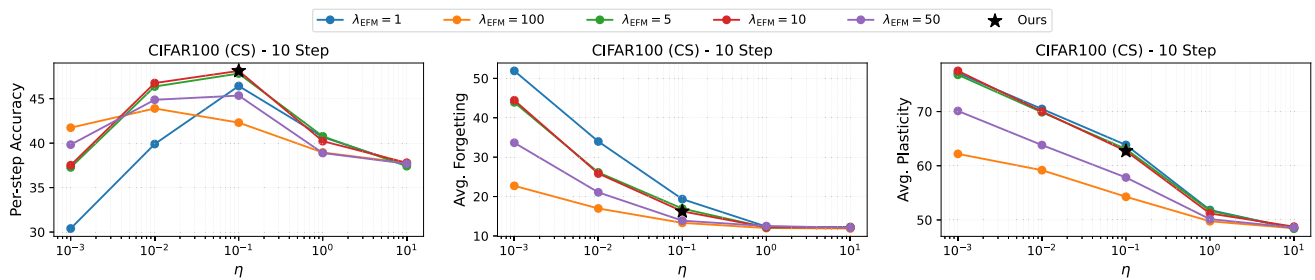


Fig. 12 A grid search testing 25 combinations of λ_{EFM} , with the x -axis in log scale for better visualization. **Left:** The selected combination, represented by the star symbol, achieves the peak per-step accuracy.

Middle: Reducing the value of η weakens regularization and increases forgetting (see Eq. 23, left). **Right:** For high values of η , plasticity is significantly compromised (see Eq. 23, right)

LwF, R-DFCIL, ABD, EFC, and EFC++ achieve comparable training times for the feature extractor, with two forward passes required for computing distillation losses in LwF, ABD and R-DFCIL, and two forward passes for computing EFM regularization in EFC and EFC++. SSRE slightly increases the training time due to architectural modifications introduced by the method. PASS is the most computationally expensive approach, as it applies self-rotation to each task, effectively doubling the training time.

In post-training, both EFC++ and R-DFCIL introduce an additional computational overhead. However, the computation of EFM and prototypes in EFC++ is negligible compared to the training time, as it only requires a single forward pass over the backbone using the current task's training dataset. Additionally, EFM can be computed without any backward pass (see Eq. 8). The prototype re-balancing phase takes approximately one-fifth of the time required for training the backbone, since the only classifier is trained and the feature extractor is frozen. In this phase, a single forward pass over the current training dataset is required to compute the task features because the feature extractor is frozen. The training time slightly increases across tasks due to the growing number of sampled prototypes.

R-DFCIL and ABD further extend training time by requiring the training of a deep generative inversion network after each task, which takes approximately 15 min – about three times longer than training the main model on CIFAR-100. This computation further increases when the resolution of the images to be generated increases (e.g. ImageNet).

Figure 13 shows the complete training times, with ABD and R-DFCIL omitted due to their significantly larger scale. Overall, when considering both its training and prototype re-balancing phases, EFC++ exhibits slightly higher training times compared to LwF and EFC, comparable timing to SSRE, and significantly lower training time than PASS. Training times were measured using an NVIDIA RTX 3090 GPU.

Memory requirements. EWC and LwF do not use prototypes, so no additional storage is required. FeTrIL, PASS,

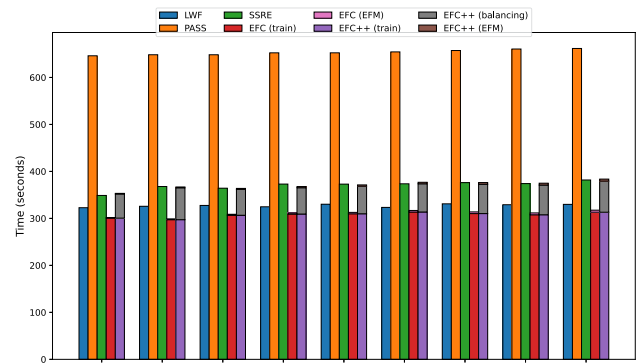


Fig. 13 Training times. For EFC++ we report the time required for backbone training, EFM and prototype computation, and the prototype re-balancing phase. Overall, EFC++ exhibits a slight increase in computational time compared to EFC and LwF, shows comparable timings to SSRE, and remains more efficient than PASS, R-DFCIL, and ABD. Timings for R-DFCIL and ABD are omitted as the deep inversion model training time alone already significantly exceeds the time required for the other approaches (about 15 min on CIFAR-100)

and SSRE need to store only class means. R-DFCIL increases storage requirements by including a generative deep inversion model with about 33 M parameters (roughly 125.89 MB at 32-bit precision). EFC, EFC++, and FeCaM must store both class means and covariances. Comparing their memory requirements to R-DFCIL's, storing 100 class covariance matrices of size 512×512 (as in CIFAR-100 with a ResNet-18 backbone) requires about 100 MB – slightly less than R-DFCIL. However, as the number of classes increases, the storage requirements for EFC, EFC++, and FeCaM grow with respect to R-DFCIL. In our previous work (Magistri et al., 2024b), we showed that these costs can be mitigated through covariance approximation, thus eliminating the need to store a covariance matrix per class—or by applying a low-rank approximation. This significantly reduces storage without significant performance degradation.

7.5 Analysis of the limitations of EFC++

Since EFC++ does not update class covariances across incremental learning steps, we aim to understand how *covariance drift* affects the final performance. Tables 7 and 8 compare different update strategies of class statistics: our approach (class means estimated via EFM and fixed covariances, highlighted in cyan) against an idealized setting using real mean and real covariances, computed from old task data fed in the current task backbone. While this strategy is infeasible in incremental learning – since past data is unavailable – it serves as an upper bound for performance and helps quantify the limitations of our prototype rehearsal strategy.

On CIFAR-100, using real class covariances provides limited benefits when the means are estimated with EFM. On ImageNet-1K, this effect is more visible, especially in the 10-step setting. Nevertheless, the dominant source of improvement is explained by the class means. Replacing

Table 7 Impact of Mean and Covariance Drift on EFC++ (CIFAR-100 Cold Start)

Step	Mean	Covariance	Accuracy	Delta
10	Fixed	Fixed	46.30 ± 1.45	–
	EFM	Fixed	47.52 ± 0.68	+1.22 (Ours)
	EFM	Real	47.37 ± 0.19	+1.07
	Real	Fixed	49.22 ± 0.17	+2.92
	Real	Real	49.33 ± 0.20	+3.03
20	Fixed	Fixed	31.71 ± 2.26	–
	EFM	Fixed	33.71 ± 1.41	+2.00 (Ours)
	EFM	Real	33.86 ± 1.49	+2.15
	Real	Fixed	37.04 ± 1.67	+5.33
	Real	Real	38.59 ± 1.64	+6.88
Joint Training			71.00	

Table 8 Impact of Mean and Covariance Drift on EFC++ (ImageNet-1K Cold Start)

Step	Mean	Covariance	Accuracy	Delta
10	Fixed	Fixed	43.65 ± 0.04	–
	EFM	Fixed	44.72 ± 0.11	+1.07 (Ours)
	EFM	Real	46.50 ± 0.27	+2.85
	Real	Fixed	49.95 ± 0.09	+6.30
	Real	Real	49.97 ± 0.09	+6.32
20	Fixed	Fixed	34.00 ± 0.03	–
	EFM	Fixed	37.46 ± 0.03	+3.46 (Ours)
	EFM	Real	38.45 ± 0.01	+4.45
	Real	Fixed	44.92 ± 0.02	+10.92
	Real	Real	45.55 ± 0.06	+11.55
Joint Training			69.76	

our estimated class means with real ones, while keeping the covariances fixed, leads to the largest improvement: 2.92% and 5.33% on CIFAR-100, and 6.30% and 10.92% on ImageNet-1K, with respect to fixed statistics in the 10- and 20-step settings, respectively. This confirms the challenges highlighted in Section 7.2 that our approach encounters when estimating the real class drift in Cold Start. Using real class covariances can further improve performance, particularly on ImageNet-1K, but this gain is much smaller than the impact of using real class means. In Appendix I, we show that similar conclusions hold for Tiny-ImageNet and ImageNet-Subset.

From this analysis, we conclude that future work on Gaussian prototypes should prioritize more effective strategies for updating class means, beyond our current EFM approach. In this direction, hybrid strategies combining the EFM pseudo metric with recent prototype update strategies such as those in LDC (Gomez-Villa et al., 2025), represents a promising direction for future work. As a final note, we report the performance under Joint Training, where the gap remains substantial. Closing this gap will likely require either a more advanced strategy than Gaussian prototypes or improved regularization techniques beyond those we proposed.

8 Conclusions

Exemplar-free Class Incremental Learning requires a careful balance between model stability and plasticity. In this paper, we tackled the challenging Cold Start scenario in which inhibiting plasticity hinders the learning of novel tasks and caused a significant drop in incremental learning accuracy. We proposed Elastic Feature Consolidation++ (EFC++), improving over EFC and achieving competitive or superior performance with respect to recent state-of-the-art methods in Exemplar-free Class Incremental Learning (EFCIL).

We derived an Empirical Feature Matrix (EFM) that induces a pseudo-metric used to control feature drift and improves plasticity with respect to the usage of feature distillation. We showed that this matrix has interesting spectral properties depending on the number of classes in the incremental learning process. This matrix is also used to update prototypes, enabling them to adapt to the feature space drift, which is particularly pronounced in the Cold Start scenario. These prototypes are leveraged in a prototype re-balancing phase to learn the inter-task classifier.

Overall, EFC++ achieves superior stability-plasticity trade-off compared to EFC across small-scale, large-scale, and domain-incremental learning benchmarks. This is achieved by applying the prototype re-balancing phase and a regularization on the feature space. Future works could develop and analyze more plastic regularizers, similar to EFM, applied to feature maps in the inner activations of the

network. The interaction between increased plasticity across different layers is still unclear.

Driven in part by the development of foundation models, recent works have begun investigating incremental learning scenarios in which the initial backbone is pre-trained on massive datasets (Li et al., 2025; Wang et al., 2026, 2025). Recent evidence shows that the benefit of pre-training depends on how well the target incremental domain is represented in the pre-training data, and that target-domain training or adaptation can remain important under significant domain shift (Feillet et al., 2025). In the future, we aim to investigate how strategies that enhance plasticity, like EFC++, perform in such pre-trained scenarios. We are particularly interested in understanding whether enhancing plasticity can improve performance when there is a large domain shift from the pre-training task, which is very common in real applications.

Acknowledgements This work was supported by ERDF/EU and Grant AIA2025-163919-C52 funded by MICIU/AEI/ 10.13039/501100011033, by the Italian national project MiSE AVIOS, by the European Union project NextGenerationEU/PRTR, and by the Spanish Development Fund FEDER.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement.

Data Availability All data employed in this paper are publicly available. In our public repository (https://github.com/simomagi/elastic_feature_consolidation), we provide detailed instructions to download and use the data for the experiments.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., & Tuytelaars, T. (2018). Memory aware synapses: Learning what (not) to forget. *Proceedings of the European conference on computer vision (ECCV)* (pp. 139–154).

Aljundi, R., Belilovsky, E., Tuytelaars, T., Charlin, L., Caccia, M., Lin, M., & Page-Caccia, L. (2019). Online continual learning with maximal interfered retrieval. In: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (eds.) *Advances in Neural Information Processing Systems*, 32. Curran Associates, Inc.

Aljundi, R., Kelchtermans, K., & Tuytelaars, T. (2019). Task-free continual learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*

Amari, S. I. (1998). Natural gradient works efficiently in learning. *Neural computation*, 10(2), 251–276.

Ayub, A., & Wagner, A. (2021). EEC: Learning to encode and regenerate images for continual learning. *International Conference on Learning Representations*

Belouadah, E., & Popescu, A. (2018). Deesil: Deep-shallow incremental learning. *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*

Belouadah, E., & Popescu, A. (2019). Il2m: Class incremental learning with dual memory. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*

Benzing, F. (2022). Unifying importance based regularisation methods for continual learning. In: G. Camps-Valls, F.J.R. Ruiz, I. Valera (eds.) *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research*, 151, (pp. 2372–2396). PMLR

Boschini, M., Bonicelli, L., Buzzega, P., Porrello, A., & Calderara, S. (2023). Class-incremental continual learning into the extended der-verse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5), 5497–5512. <https://doi.org/10.1109/TPAMI.2022.3206549>

Buzzega, P., Boschini, M., Porrello, A., Abati, D., & Calderara, S. (2020). Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33, 15920–15930.

Caselli, L., Magistri, S., Bianconcini, T., Benericetti, A., de Andrade, D. C., & Bagdanov, A. D. (2025). Incremental and decremental continual learning for privacy-preserving video recognition. In A. Del Bue, C. Canton, J. Pont-Tuset, & T. Tommasi (Eds.), *Computer Vision - ECCV 2024 Workshops* (pp. 333–350). Cham: Springer Nature Switzerland.

Castro, F. M., Marín-Jiménez, M. J., Guil, N., Schmid, C., & Alahari, K. (2018). End-to-end incremental learning. *Proceedings of the European conference on computer vision (ECCV)* (pp. 233–248).

Chaudhry, A., Dokania, P. K., Ajanthan, T., & Torr, P. H. (2018). Riemannian walk for incremental learning: Understanding forgetting and intransigence. *Proceedings of the European conference on computer vision (ECCV)* (pp. 532–547)

Davari, M., Asadi, N., Mudur, S., Aljundi, R., & Belilovsky, E. (2022). Probing representation forgetting in supervised and unsupervised continual learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2022*

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., & Tuytelaars, T. (2022). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3366–3385. <https://doi.org/10.1109/TPAMI.2021.3057446>

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition, (pp. 248–255). Ieee

Deng, J., Wu, Q., Ju, P., Lin, S., Liang, Y., & Shroff, N. (2025). Unlocking the power of rehearsal in continual learning: A theoretical perspective. *Forty-second International Conference on Machine Learning*

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houthby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*

Douillard, A., Cord, M., Ollion, C., Robert, T., & Valle, E. (2020). Podnet: Pooled outputs distillation for small-tasks incremental learning. In: *Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, (pp. 86–102). Springer

- Feillet, E., Popescu, A., & Hudelot, C. (2025). A reality check on pre-training for exemplar-free class-incremental learning. *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)* (pp. 7614–7625).
- French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4), 128–135. [https://doi.org/10.1016/S1364-6613\(99\)01294-2](https://doi.org/10.1016/S1364-6613(99)01294-2)
- Gao, Q., Zhao, C., Ghanem, B., & Zhang, J. (2022). R-dfcil: Relation-guided representation learning for data-free class incremental learning. *European Conference on Computer Vision* (pp. 423–439). Springer.
- Gomez-Villa, A., Goswami, D., Wang, K., Bagdanov, A. D., Twardowski, B., & van de Weijer, J. (2025). Exemplar-free continual representation learning via learnable drift compensation. In A. Leonardis, E. Ricci, S. Roth, O. Russakovsky, T. Sattler, & G. Varol (Eds.), *Computer Vision - ECCV 2024* (pp. 473–490). Cham: Springer Nature Switzerland.
- Goswami, D., Liu, Y., Twardowski, B., & van de Weijer, J. (2023). FeCAM: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. In: Thirty-seventh Conference on Neural Information Processing Systems
- Goswami, D., Soutif-Cormerais, A., Liu, Y., Kamath, S., Twardowski, B., & Van De Weijer, J. (2024). Resurrecting Old Classes with New Data for Exemplar-Free Continual Learning. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), (pp. 28525–28534). IEEE Computer Society, Los Alamitos, CA, USA. <https://doi.org/10.1109/CVPR52733.2024.02695>. <https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.02695>
- Gowda, S., Zonooz, B., & Arani, E. (2023). Dual cognitive architecture: Incorporating biases and multi-memory systems for lifelong learning. *Transactions on Machine Learning Research*.
- Grementieri, L., & Fioresi, R. (2022). Model-centric data manifold: the data through the eyes of the model. *SIAM Journal on Imaging Sciences*, 15(3), 1140–1156.
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770–778)
- He, R., Fang, D., Chen, Y., Tong, K., Chen, C., Wang, Y., pui Chau, L., & Zhuang, H. (2026). Real: Representation enhanced analytic learning for exemplar-free class-incremental learning. *Knowledge-Based Systems*, 332, Article 114901. <https://doi.org/10.1016/j.knosys.2025.114901>
- He, R., Fang, D., Xu, Y., Cui, Y., Li, M., Chen, C., Zeng, Z., & Zhuang, H. (2025). Semantic shift estimation via dual-projection and classifier reconstruction for exemplar-free class-incremental learning. Forty-second International Conference on Machine Learning
- Hess, T., Verwimp, E., van de Ven, G.M., & Tuytelaars, T. (2024). Knowledge accumulation in continually learned representations and the issue of feature forgetting. *Transactions on Machine Learning Research*
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network [arXiv:1503.02531](https://arxiv.org/abs/1503.02531) arXiv preprint.
- Hou, S., Pan, X., Loy, C. C., Wang, Z., & Lin, D. (2019). Learning a unified classifier incrementally via rebalancing. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 831–839)
- Huszár, F. (2018). Note on the quadratic penalties in elastic weight consolidation. *Proceedings of the National Academy of Sciences*, 115(11), E2496–E2497.
- Jiang, S., Fang, Y., Zhang, H., Liu, Q., Qi, Y., Yang, Y., & Wang, P. (2026). Teacher agent: A knowledge distillation-free framework for rehearsal-based video incremental learning. *International Journal of Computer Vision*, 134(4), 190. <https://doi.org/10.1007/s11263-026-02779-2>
- Jung, H., Ju, J., Jung, M., & Kim, J. (2016). Less-forgetting learning in deep neural networks. arXiv preprint [arXiv:1607.00122](https://arxiv.org/abs/1607.00122)
- Kang, H., Seifer, G., Lee, D., & Ryu, J. (2025). Do your best and get enough rest for continual learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 10077–10086)
- Kang, M., Park, J., & Han, B. (2022). Class-incremental learning by knowledge distillation with adaptive feature consolidation. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 16071–16080)
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13), 3521–3526.
- Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images Technical report.
- Lange, M.D., van de Ven, G.M., & Tuytelaars, T. (2023). Continual evaluation for lifelong learning: Identifying the stability gap. In: The Eleventh International Conference on Learning Representations
- Lanzillotta, G., Meier, D., & Hofmann, T. (2026). Heads collapse, features stay: Why replay needs big buffers. The Fourteenth International Conference on Learning Representations
- Lee, C. S., & Lee, A. Y. (2020). Clinical applications of continual learning machine learning. *The Lancet Digital Health*, 2(6), e279–e281. [https://doi.org/10.1016/S2589-7500\(20\)30102-3](https://doi.org/10.1016/S2589-7500(20)30102-3)
- Li, L., Zhou, D. W., Ye, H. J., & Zhan, D. C. (2025). Addressing imbalanced domain-incremental learning through dual-balance collaborative experts. Forty-second International Conference on Machine Learning
- Li, Z., & Hoiem, D. (2017). Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12), 2935–2947.
- Liu, M., Ma, J., Kang, Q., Zhou, M., & Albeshri, A. (2026). Exemplar-free class incremental learning for traffic sign classification. *IEEE Transactions on Vehicular Technology*, 75(3), 3556–3568. <https://doi.org/10.1109/TVT.2025.3606740>
- Liu, W., Nie, X., Zhang, B., & Sun, X. (2022). Incremental learning with open-set recognition for remote sensing image scene classification. *IEEE Transactions on Geoscience and Remote Sensing*, 60, 1–16. <https://doi.org/10.1109/TGRS.2022.3173995>
- Liu, X., & Chang, X. (2026). Elastic weight consolidation done right for continual learning. *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*
- Liu, X., Masana, M., Herranz, L., Van de Weijer, J., Lopez, A.M., & Bagdanov, A.D. (2018). Rotate your networks: Better weight consolidation and less catastrophic forgetting. In: 2018 24th International Conference on Pattern Recognition (ICPR), (pp. 2262–2268). IEEE
- Liu, X., Wu, C., Menta, M., Herranz, L., Raducanu, B., Bagdanov, A.D., Jui, S., & de Weijer, J.v. (2020). Generative feature replay for class-incremental learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, (pp. 226–227)
- Liu, Y., Schiele, B., & Sun, Q. (2021). Adaptive aggregation networks for class-incremental learning. *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition* (pp. 2544–2553)
- Lopez-Paz, D., & Ranzato, M. A. (2017). Gradient episodic memory for continual learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 30., Curran Associates Inc.
- Magistri, S., Baracchi, D., Shullani, D., Bagdanov, A. D., & Piva, A. (2024). Continual learning for adaptive social network identification. *Pattern Recognition Letters*, 180, 82–89. <https://doi.org/10.1016/j.patrec.2024.02.020>

- Magistri, S., Trinci, T., Soutif, A., van de Weijer, J., & Bagdanov, A.D. (2024). Elastic feature consolidation for cold start exemplar-free incremental learning. In: The Twelfth International Conference on Learning Representations
- Martens, J. (2020). New insights and perspectives on the natural gradient method. *The Journal of Machine Learning Research*, 21(1), 5776–5851.
- Martens, J., & Sutskever, I. (2012). Training deep and recurrent networks with hessian-free optimization. *Neural Networks: Tricks of the Trade: Second Edition* (pp. 479–535). Springer.
- Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D., & Van De Weijer, J. (2022). Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5), 5513–5533.
- McCloskey, M., Cohen, N.J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In: Psychology of learning and motivation, 24, (pp. 109–165). Elsevier
- Mermillod, M., Bugajska, A., & BONIN, P. (2013). The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in Psychology*, 4, <https://doi.org/10.3389/fpsyg.2013.00504>
- Mittal, S., Galesso, S., & Brox, T. (2021). Essentials for class incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (pp. 3513–3522)
- Nguyen, X. B., Serna-Aguilera, M., Choudhary, A. K., Sinha, P., Li, X., & Luu, K. (2026). COBRA: A continual learning approach to vision-brain understanding. *International Journal of Computer Vision*, 134(1), 30. <https://doi.org/10.1007/s11263-025-02617-x>
- Petit, G., Popescu, A., Schindler, H., Picard, D., & Delezoide, B. (2023). Fetril: Feature translation for exemplar-free class-incremental learning. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 3911–3920)
- Qazi, M.A., Hashmi, A.U.R., Sanjeev, S., Almakky, I., Saeed, N., González, C., & Yaqub, M. (2026). Continual learning in medical imaging: A survey and practical analysis. *ACM Comput. Surv.* 58(8). <https://doi.org/10.1145/3785663>
- Rebuffi, S. A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 2001–2010)
- Ritter, H., Botev, A., & Barber, D. (2018). Online structured laplace approximations for overcoming catastrophic forgetting. *Advances in Neural Information Processing Systems* (p. 31)
- Rypeś, G., Cygert, S., Trzciński, T., & Twardowski, B.o. (2024). Task-recency bias strikes back: Adapting covariances in exemplar-free class incremental learning. In: A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, C. Zhang (eds.) *Advances in Neural Information Processing Systems*, 37, (pp. 63268–63289). Curran Associates, Inc. <https://doi.org/10.52202/079017-2021>. https://proceedings.neurips.cc/paper_files/paper/2024/file/73ba81c7b25134a559c8a9c39ec1a4c3-Paper-Conference.pdf
- Shaheen, K., Hanif, M. A., Hasan, O., & Shafique, M. (2022). Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks. *Journal of Intelligent & Robotic Systems*, 105(1), 9. <https://doi.org/10.1007/s10846-022-01603-6>
- Shin, H., Lee, J.K., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc.
- Smith, J., Hsu, Y. C., Balloch, J., Shen, Y., Jin, H., & Kira, Z. (2021). Always be dreaming: A new approach for data-free class-incremental learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 9374–9384)
- Soutif-Cormerais, A., Carta, A., Cossu, A., Hurtado, J., Lomonaco, V., Van de Weijer, J., & Hemati, H. (2023). A comprehensive empirical evaluation on online continual learning. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, (pp. 3518–3528)
- Toldo, M., & Ozay, M. (2022). Bring evanescent representations to life in lifelong class incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 16732–16741)
- Trinci, T., Magistri, S., Verdecchia, R., & Bagdanov, A. D. (2025). How green is continual learning, really? analyzing the energy consumption in continual training of vision foundation models. In A. Del Bue, C. Canton, J. Pont-Tuset, & T. Tommasi (Eds.), *Computer Vision - ECCV 2024 Workshops* (pp. 300–317). Cham: Springer Nature Switzerland.
- van de Ven, G.M., Tuytelaars, T., & Tolias, A.S. (2022) Three types of incremental learning. *Nature Machine Intelligence* 4(12), 1185–1197. <https://doi.org/10.1038/s42256-022-00568-3>
- Verwimp, E., Aljundi, R., Ben-David, S., Bethge, M., Cossu, A., Geperth, A., Hayes, T.L., Hüllermeier, E., Kanan, C., Kudithipudi, D., Lampert, C.H., Mundt, M., Pascanu, R., Popescu, A., Tolias, A.S., van de Weijer, J., Liu, B., Lomonaco, V., Tuytelaars, T., & van de Ven, G.M. (2024). Continual learning: Applications and the road forward. *Transactions on Machine Learning Research*
- Vogelstein, J. T., Dey, J., Helm, H. S., LeVine, W., Mehta, R. D., Tomita, T. M., Xu, H., Geisa, A., Wang, Q., van de Ven, G. M., Gao, C., Yang, W., Tower, B., Larson, J., White, C. M., & Priebe, C. E. (2025). Simple lifelong learning machines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(11), 10033–10046. <https://doi.org/10.1109/TPAMI.2025.3595364>
- Wang, C., Jiang, J., Hu, X., Liu, X., & Ji, X. (2025). Enhancing consistency and mitigating bias: A data replay approach for incremental learning. *Neural Networks*, 184, Article 107053. <https://doi.org/10.1016/j.neunet.2024.107053>
- Wang, H., Cheng, D., He, L., Li, Y., Li, J., Wang, N., & Gao, X. (2026). EKPC: Elastic knowledge preservation and compensation for class-incremental learning. *International Journal of Computer Vision*, 134(5), 238. <https://doi.org/10.1007/s11263-026-02820-4>
- Wang, L., Zhang, X., Su, H., & Zhu, J. (2024). A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (pp. 1–20). <https://doi.org/10.1109/TPAMI.2024.3367329>
- Wang, Y., Zhou, D. W., & Ye, H. J. (2025). Integrating task-specific and universal adapters for pre-trained model-based class-incremental learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 806–816)
- Wu, J., Zhang, Q., & Xu, G. (2017). Tiny imagenet challenge Technical report.
- Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., & Fu, Y. (2019). Large scale incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*
- Xiang, Y., Fu, Y., Ji, P., & Huang, H. (2019). Incremental learning using conditional adversarial networks. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*
- Yan, S., Xie, J., & He, X. (2021). Der: Dynamically expandable representation for class incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 3014–3023)
- Yu, L., Twardowski, B., Liu, X., Herranz, L., Wang, K., Cheng, Y., Jui, S., & Weijer, J.v.d. (2020). Semantic drift compensation for class-incremental learning. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (pp. 6982–6991)
- Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. In: D. Precup, Y.W. Teh (eds.) *Proceedings of the 34th International Conference on Machine Learning, Pro-*

- ceedings of Machine Learning Research*, 70, (pp. 3987–3995). PMLR
- Zhang, Y., Pfahringer, B., Frank, E., Bifet, A., Lim, N. J. S., & Jia, Y. (2022). A simple but strong baseline for online continual learning: Repeated augmented rehearsal. *Advances in Neural Information Processing Systems*, 35, 14771–14783.
- Zhou, D. W., Sun, H. L., Ning, J., Ye, H. J., & Zhan, D. C. (2024). Continual learning with pre-trained models: A survey. *IJCAI*, 8363–8371.
- Zhou, D. W., Wang, F. Y., Ye, H. J., & Zhan, D. C. (2023). Pycil: a python toolbox for class-incremental learning. *SCIENCE CHINA Information Sciences*, 66(9), Article 197101. <https://doi.org/10.1007/s11432-022-3600-y>
- Zhou, D. W., Wang, Q. W., Qi, Z. H., Ye, H. J., Zhan, D. C., & Liu, Z. (2023). Deep class-incremental learning: A survey [arXiv:2302.03648](https://arxiv.org/abs/2302.03648) arXiv preprint.
- Zhou, D. W., Wang, Q. W., Qi, Z. H., Ye, H. J., Zhan, D. C., & Liu, Z. (2024). Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12), 9851–9873.
- Zhou, D. W., Wang, Q. W., Ye, H. J., & Zhan, D. C. (2022). A model or 603 exemplars: Towards memory-efficient class-incremental learning. The Eleventh International Conference on Learning Representations
- Zhu, F., Cheng, Z., Zhang, X. y., & Liu, C. l. (2021). Class-incremental learning via dual augmentation. *Advances in Neural Information Processing Systems*, 34, 14306–14318.
- Zhu, F., Zhang, X. Y., Wang, C., Yin, F., & Liu, C. L. (2021). Prototype augmentation and self-supervision for incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5871–5880)
- Zhu, K., Zhai, W., Cao, Y., Luo, J., & Zha, Z. J. (2022). Self-sustaining representation expansion for non-exemplar class-incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 9296–9305)
- Zhuang, H., Weng, Z., Wei, H., XIE, R., Toh, K. A., & Lin, Z. (2022). ACIL: Analytic class-incremental learning with absolute memorization and privacy protection. In: A. H. Oh, A. Agarwal, D. Belgrave, K. Cho (eds.) *Advances in Neural Information Processing Systems*. <https://openreview.net/forum?id=Vc4QUfqr4do>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.