# Debiasing Few-Shot Class-Incremental Learning via Dynamic Feature-Classifier Alignment

**Kim, Shiwon**

**Department of Digital Analytics**

**Graduate School**

**Yonsei University**

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# ABSTRACT

## Debiasing Few-Shot Class-Incremental Learning via Dynamic Feature-Classifier Alignment

Few-shot class-incremental learning (FSCIL) addresses the problem of continuously learning new classes from only a few training samples, while retaining knowledge of previously learned ones. In this setting, the severe imbalance between base and incremental classes often leads to a strong bias toward the base classes and limited adaptability to new classes. To tackle this issue, recent approaches leverage the geometric properties of the simplex equiangular tight frame (ETF), which emerges in the terminal phase of deep model training as part of the neural collapse phenomenon. However, existing ETF-based methods assume a fixed classifier determined at the base session with a pre-defined number of classes, which is impractical in FSCIL where new classes are continually added. In this study, we propose a Dynamic Hierarchical ETF (DH-ETF) classifier that incorporates semantic similarity among classes to achieve both scalability and discriminability. After training on the base session, we construct a hierarchy map by clustering the base class prototypes and build a hierarchical ETF classifier composed of a fixed cluster-level ETF and adaptable per-cluster class-level ETFs. In each incremental session, the class-level ETFs are updated based on the number of new classes and their similarity to the base classes in the embedding space. This design preserves prior knowledge through the fixed cluster-level representation while facilitating adaptation to new classes via the updated class-level ETFs guided by the hierarchy. Experiments on CIFAR-100 show that our method outperforms existing state-of-the-art approaches, effectively addressing base-class bias by achieving superior incremental class performance while alleviating catastrophic forgetting.

**Key words** : Few-shot class-incremental learning, Neural collapse, Equiangular tight frame (ETF)

# 1. Introduction

## 1.1. Research Background

Deep neural networks (DNNs) have achieved remarkable success across a range of real-world domains, often matching or even surpassing human-level performance [1-3]. However, when trained on data streams, they suffer from *catastrophic forgetting* [4, 5], which refers to losing previously acquired knowledge while adapting to evolving data distributions. To tackle this challenge, class-incremental learning (CIL) has been introduced as a deep learning framework that enables models to continuously accommodate new classes over time while preserving knowledge from previously observed ones [6, 7]. While CIL has been widely studied due to its great potential in dynamically incorporating new classes, it tends to be less effective in scenarios where only a small number of new samples are available. This type of scenario commonly arises in real-world contexts, such as diagnosing rare diseases in the medical field [8, 9] or recognizing unseen objects in autonomous driving [10]. Inspired by such practical constraints, the few-shot class-incremental learning (FSCIL) paradigm has emerged as a more realistic yet challenging extension of CIL, under the assumption that new classes are introduced with only a few samples [11, 12]. Specifically, the FSCIL task consists of a base session that includes a large number of classes with sufficient training samples per class, followed by multiple incremental sessions where an extremely limited number of samples are provided [13]. Such imbalance in FSCIL poses a unique challenge beyond catastrophic forgetting, as the model's performance tends to be biased toward the base classes.

FSCIL has gained much attention due to its practical importance and challenging nature. Early approaches primarily focus on maintaining the discriminability of previous classes when training on new data. These methods are referred to as *backward compatible* FSCIL [13, 14], which typically finetune the backbone network with knowledge distillation schemes to reduce the forgetting [11, 15], or freeze the backbone trained on base classes and update only the projection layer or classifier head during incremental sessions [16, 17]. However, these methods still suffer from limited adaptability to newly introduced classes, as the model tends to retain a strong bias toward the base classes and struggles to effectively incorporate new information. A better solution is to anticipate future updates in advance when training the early version of a model. Such forward compatibility is regarded a more desirable property than backward compatibility for FSCIL. *Forward compatible* approaches aim to prepare for possible future updates during base session training, by learning a more compact embedding space for base classes to reserve space for new classes [18-21].

Among forward compatible methods, one notable line of work employs a simplex equiangular tight frame (ETF) classifier [21], inspired by the recently discovered *Neural Collapse* phenomenon.

Neural collapse refers to the behavior of deep models at the terminal phase of training, where the last-layer features of each class collapse into a single vertex aligned with their respective class means (*i.e.,* prototypes), which together form a simplex ETF [22]. A simplex ETF is a geometric structure in which all vectors are equiangular, thereby maximizing the pair-wise angles among these vectors. [21] adopts an ETF classifier in FSCIL to construct a well-structured feature space that facilitates the seamless addition of new classes. It is designed to reserve balanced space for future classes by maximizing the inter-class distance while maintaining equal angles between class prototypes.

## 1.2. Motivations and Contributions

The ETF classifier has emerged as a powerful tool, achieving state-of-the-art performance in scenarios where training data for certain classes is limited. However, in this study, we focus more on the practical applicability of the ETF classifier in class-incremental settings. [21] pre-defines an ideal set of directions for class-wise features using the ETF classifier during the base session, and subsequently aligns new class features to these directions in later sessions without adjusting the classifier weights. Even though such strategy proves to be effective in minimizing representation shifts, this "fixed" classifier design raises a fundamental question: *is it appropriate to use a classifier pre-assigned in the base session when new classes continue to arrive?*

Figure 1 illustrates the average accuracy of [21] on CIFAR-100 [23], with the ETF classifier dimensionality K varying from 60 to 150. When $K < 100$, the total number of classes eventually exceeds K, making the classifier unable to predict additional classes beyond K. Therefore, we set the accuracy of these classes to zero. The model reaches its highest performance when K equals the total number of classes (*i.e.,* $K = 100$), while the accuracy drops sharply for $K < 100$ and gradually declines for $K > 100$. These results indicate that FSCIL with a pre-defined ETF classifier is optimal only when the number of classes in the dataset is specified in advance. However, this assumption is impractical in class-incremental scenarios, where new classes continuously emerge over time.

A recent study highlights the infeasibility of a fixed ETF classifier in class-incremental settings, and proposes *dynamic neural collapse* in the context of online task-free continual learning [24]. In each incremental phase, the ETF classifier is recomputed based on the old and new class prototypes to reflect the increased number of classes. While this approach is practical and promising in that it allows continuous adjustment of the ETF classifier during incremental training, it also introduces a trade-off: it may exacerbate forgetting of previously observed classes and lead to overfitting to the newly added classes, which can undermine the overall discriminability between classes.

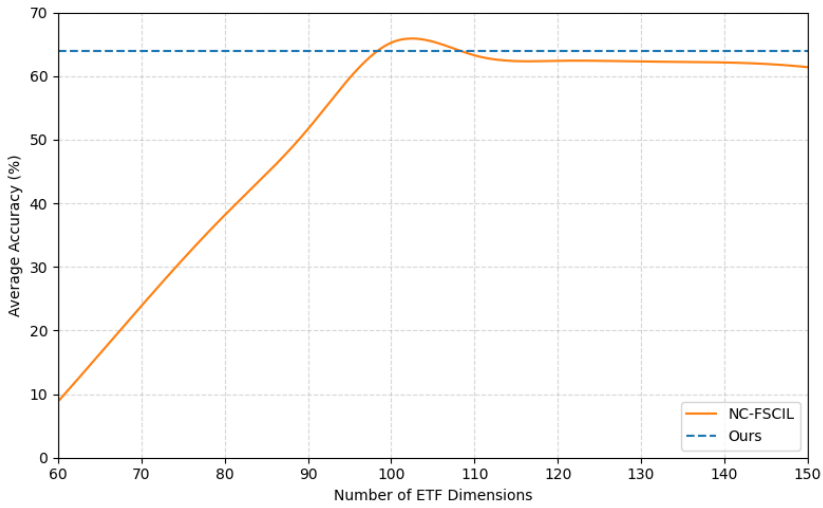



**Figure 1.** Performance on CIFAR-100 with ETF dimensions from 60 to 150. Varying dimensions are only applied to NC-FSCIL which requires a fixed ETF classifier.

In this study, we leverage hierarchical information derived from the *semantic similarity* among classes to construct a scalable ETF classifier while compensating for the performance degradation that may arise from updating the classifier weights. The use of hierarchical information to improve the performance of deep classification models has been extensively studied across various domains, including medical [25-27] and e-commerce [28, 29] applications. Moreover, it has also demonstrated its effectiveness in imbalanced learning [30-32] and continual learning [1, 33]. Using a hierarchical structure allows the classification problem to be addressed in stages—solve a less complex, higher-level task first, and utilize its resulting information to guide a more fine-grained, lower-level task. Inspired by this insight, we aim to strike a balance between the scalability and performance of the ETF classifier by decomposing it into two levels, each assigned with a distinct role. After training on the base session, we construct a semantic hierarchy map by clustering the prototypes of the base classes. When a new class arrives, it is assigned to a class cluster based on its semantic similarity to the cluster centroids, and the hierarchy map is updated accordingly. This hierarchy map is used to build a dual-level ETF classifier consisting of a cluster-level and a class-level component. The cluster-level ETF classifier is constructed as a (K+1)-dimensional ETF with an additional out-of-distribution (OOD) cluster, while the class-level ETF classifier comprises K distinct ETF structures, each designed to separate the specific classes within its corresponding cluster. During incremental sessions, we finetune only the dual-level projection layers appended to the backbone, which drive the last layer features at each level toward their respective target ETF prototypes. The training is regularized by a dependency term, which guides the model to make predictions consistent with the

hierarchical structure. The cluster-level ETF classifier remains fixed while only the class-level ETF classifier is recalculated as new classes arrive.

This hierarchy-aware design enhances the distinction between semantically similar few-shot categories by enabling fine-grained differentiation at the class level within each cluster. It also shifts the burden of maintaining priorly acquired knowledge to the cluster-level features through a *fixed* cluster-level ETF classifier, while continually adapting to newly introduced classes by aligning their class-level features to the *updated* class-level ETF classifier. Our proposed *Dynamic Hierarchical ETF (DH-ETF)* classifier not only effectively prevents catastrophic forgetting, but also mitigates the inherent bias of FSCIL by achieving superior incremental class performance compared to other state-of-the-art methods. The contributions of this paper are summarized as follows:

- We propose a novel Dynamic Hierarchical ETF (DH-ETF) classifier that leverages class-wise semantic hierarchy to achieve both scalability and discriminability in FSCIL.
- We demonstrate through extensive experiments that our method outperforms other state-of-the-art methods, addressing the base class bias by achieving superior incremental class performance while effectively alleviating catastrophic forgetting.

# 2. Related Work

**Few-Shot Class-Incremental Learning (FSCIL).** As an extended variant of conventional class-incremental learning (CIL), FSCIL addresses the challenge of continually adapting to new classes using only a few samples per class while maintaining previously acquired knowledge. The FSCIL scenario is first introduced by TOPIC [11], which employs the neural gas (NG) structure to model the topology between the base and incremental classes. Recent efforts can be broadly categorized into *backward compatible* and *forward compatible* approaches. Backward compatible approaches primarily focus on preserving the knowledge of previously observed classes, typically by adopting distillation schemes when finetuning the backbone network [15], or by freezing the backbone and updating only the classifier [16, 17]. On the other hand, forward compatible FSCIL methods aims to prepare for possible future class updates during base training. A recent forward compatible work studies FSCIL from the perspective of the widely discussed *neural collapse* phenomenon [21]. It utilizes the simplex equiangular tight frame (ETF) structure to create a stable incremental classifier for FSCIL. It achieves *state-of-the-art* performance by fixing the classifier weights to the optimal symmetric ETF configuration and aligning the features of new classes to this structure. However, it lacks scalability since its classifier weights are pre-assigned at the initial training stage and frozen

throughout incremental training. In contrast, our method continually updates the ETF classifier to adapt to the number of new classes in each session. We mitigate the representation shift caused by adjusting the classifier weights by leveraging class semantics and hierarchy.

**Neural Collapse.** Neural collapse is a phenomenon observed at the terminal phase of training deep neural networks for classification tasks, particularly when models are trained to zero training error using cross-entropy loss and mean-squared error functions [22]. As a result of this phenomenon, the class-wise means of the last-layer features collapse to an optimal geometric structure known as the simplex equiangular tight frame (ETF). Recent studies have demonstrated that a simple last-layer optimization based on the ETF classifier delivers strong generalization performance across different learning paradigms, including imbalanced learning [34-36], transfer learning [37, 38]. In particular, it has been adopted in various continual learning scenarios [21, 24, 39], such as class-incremental learning, domain-incremental learning, and online continual learning. NC-FSCIL [21] extends this idea to the few-shot class-incremental learning (FSCIL) setting, where only a few samples per class are available for new classes. It pre-defines an ETF classifier and subsequently aligns the new class prototypes to this fixed structure. However, due to its reliance on a "fixed" classifier, it is hard to achieve optimal performance without knowing the total number of classes encountered throughout the incremental process. In response, DYSON [24] proposes to dynamically recompute the ETF structure with new class prototypes in the context of online continual learning. A recent approach in domain-incremental learning (DIL) aligns image features from varying image domains to a fixed dual-level ETF classifier constructed from the text embeddings of class names [39]. Our method is inspired by [39], but we dynamically update the hierarchical ETF classifier in a class-incremental fashion. To the best of our knowledge, we are the first to leverage class semantics in FSCIL from the neural collapse perspective.

# 3. Preliminaries

## 3.1. Few-Shot Class-Incremental Learning

The few-shot class-incremental learning (FSCIL) task aims to adapt a model to a continuously expanding label space using only a few labeled samples for each newly introduced classes. The task begins with a base session including ample training data, followed by multiple incremental sessions

where only a limited number of training samples are available. In each session, the model is trained solely on the data pertinent to that session, with no access to data from previous sessions.

FSCIL trains a model incrementally on a sequence of training datasets $\{\mathcal{D}^{(0)}, \mathcal{D}^{(1)}, \dots, \mathcal{D}^{(T)}\}$, where $\mathcal{D}^{(t)} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}^{(t)}|}$, $\mathcal{D}^{(0)}$ is the base session, and T is the number of incremental sessions. The base session $\mathcal{D}^{(0)}$ contains a large label space and sufficient training images for each class. In each incremental session $\mathcal{D}^{(t)}$ for $t > 0$, there are only a few labeled images and we have $|\mathcal{D}^{(t)}| = nk$, where $n$ is the number of classes and $k$ is the number of samples per class, known as $n$-way $k$-shot. The label space in each session does not overlap with any other session. For evaluation in the $t$-th session, the label space of the test dataset consists of all the encountered classes in the previous and current sessions.

## 3.2. Simplex Equiangular Tight Frame

A simplex equiangular tight frame (ETF) is a geometric structure formed at the terminal phase of training, which maximizes the pair-wise angles of all vectors. Specifically, it refers to a matrix composed of $K$ vectors $E = [e_1, \dots, e_K] \in \mathbb{R}^{d \times K}$ that satisfies:

$$E = \sqrt{\frac{K}{K-1}} U \left( I_K - \frac{1}{K} 1_K 1_K^\top \right)$$

where $U \in \mathbb{R}^{d \times K}$ is an orthogonal matrix that allows rotation and satisfies $U^\top U = I_K$, $I_K$ is the identity matrix, and $1_K$ is an all-ones vector. All column vectors in $E$ have the same $\ell_2$ norm and any pair satisfies:

$$e_i{}^\top e_j = \begin{cases} 1 & \text{if } i = j \\ -\frac{1}{K-1} & \text{if } i \neq j \end{cases}$$

The pair-wise angle $-\frac{1}{K-1}$ is the maximal equiangular separation of $K$ vectors in $\mathbb{R}^{d \times K}$.

## 3.3. Dual-Level ETF Classifier

The concept of dual-level ETF classifier is first introduced by DualCP [39] in the context of domain-incremental learning (DIL). In DIL scenarios, image features vary across domains while the

semantic meaning of classes remains invariant. To this end, DualCP leverages the domain-invariant text embedding space as an anchor. Specifically, it constructs a fixed hierarchical ETF classifier in the text space, which is assumed to be stable across domains, and aligns the varying image features from different domains to this anchor space. The text embedding space remains invariant throughout sessions, enabling consistent alignment even as the image space changes. This facilitates domain-invariant representation learning by decoupling domain-specific variations from class semantics.

However, applying this approach directly to class-incremental learning (CIL) is suboptimal, as CIL introduces new classes over time and the text embedding space may not adequately reflect the evolving class distribution. In CIL, both the image and text feature spaces are subject to change as new classes are added. Therefore, maintaining a fixed text-based anchor can introduce noise rather than providing meaningful guidance, and can even exacerbate forgetting in the image space. Instead, it becomes crucial to construct a hierarchy-aware classifier that dynamically captures the semantic relationships among newly added classes while preserving meaningful inter-class separability in the image feature space.

Our method tackles this limitation by dynamically creating a hierarchical ETF classifier based on the semantics of class prototypes extracted from the image space itself. Unlike DualCP, which fixes the text embeddings as anchors and aligns the image features to them, we leverage the semantic hierarchy among the image prototypes to create a more adaptable and meaningful image classifier that evolves as new classes are added. This design enables the classifier to maintain semantically consistent representation, while avoiding the risk of noise from misaligned anchors. Furthermore, when training, we incorporate a dependency loss term to enforce consistency between the cluster-level and class-level predictions, which explicitly encodes the hierarchical structure in the projection layers and allows for a more robust feature space.

# 4. Methods

Figure 2 depicts the overall framework of our proposed Dynamic Hierarchical ETF classifier (DH-ETF) approach. Our method consists of two main components: (a) *Neural Collapse-Inspired Hierarchical Label Construction*, which constructs a hierarchical ETF classifier based on semantic similarity among classes, and (b) *Dynamic Hierarchical Feature-Classifier Alignment*, which trains dual-level projection layers using the updated hierarchical ETF classifier. Section 4.1. describes how the hierarchical ETF is constructed in the base session, and dynamically updated during incremental sessions. Section 4.2. presents the hierarchical loss function used to train each projection layer.

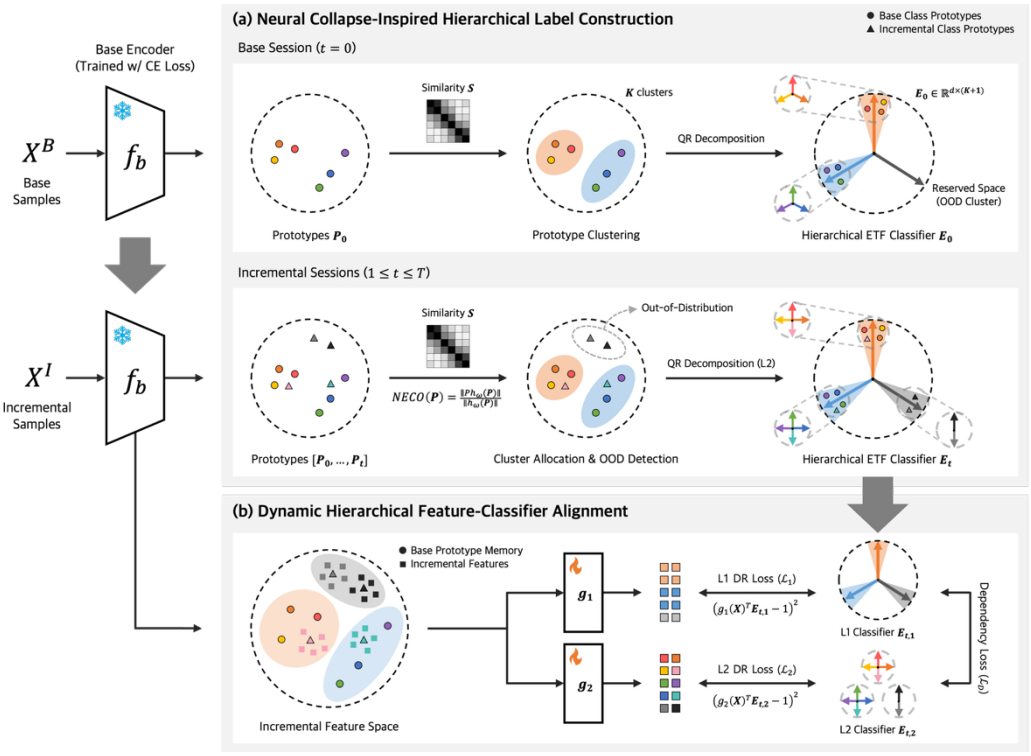



**Figure 2.** The overall framework of Dynamic Hierarchical ETF (DH-ETF).

## 4.1. Neural Collapse-Inspired Hierarchical Label Construction

This section describes how the hierarchical ETF is initially constructed during the base session and how it is updated in the incremental sessions. In the base session, we first extract the base class prototypes using the trained encoder, then construct the cluster-level and class-level ETF classifiers based on the clustering results of these prototypes. In each incremental session, the class hierarchy is updated based on the semantic similarity between the prototypes of newly introduced classes and the existing cluster centroids. The class-level ETF classifier is subsequently updated according to the revised hierarchy.

### 4.1.1. Base Session

**Prototype Clustering.** To extract meaningful features for the base classes, we first train the base encoder on the base class training data using the cross-entropy (CE) loss. With the trained encoder, we compute the class-wise means, referred to as prototypes, for each base class. These prototypes are then used to construct a semantic hierarchy map of the classes through clustering. The clustering is performed based on a similarity matrix $\boldsymbol{S}$, which is computed by calculating the pair-wise cosine similarities between all base prototypes $\boldsymbol{P}_0$. Classes whose cosine similarity exceeds a predefined hyperparameter $p$ are grouped into the same class cluster. This process results in $\boldsymbol{K}$ clusters.

**Out-of-Distribution Cluster Initialization.** To handle incremental classes significantly different from the existing base classes, we pre-define an out-of-distribution (OOD) cluster to reserve space for such OOD classes in advance. We first compute the centroids of the $\boldsymbol{K}$ clusters obtained through prototype clustering $\boldsymbol{C} = [c_0, \dots, c_K]$, and assign a vector $o$ that is orthogonal to all $\boldsymbol{K}$ centroids as the centroid of the OOD cluster. The $\boldsymbol{K}$+1 centroid vectors $\boldsymbol{C}' = [\boldsymbol{C}; o]$ including $\boldsymbol{K}$ in-distribution (ID) cluster centroids $\boldsymbol{C}$ and a OOD cluster centroid $o$ are used to construct a ($\boldsymbol{K}$+1)-dimensional cluster-level ETF classifier.

**Hierarchical ETF Construction.** We construct the hierarchical ETF classifier using the cluster centroids $\boldsymbol{C}' = [c_0, \dots, c_K, o]$ and base prototypes $\boldsymbol{P}_0$. The ETF structure is derived by performing QR decomposition on a specific matrix to obtain an orthogonal matrix $\boldsymbol{Q}$ [34]. To build the dual-level ETF, we perform two separate ETF construction processes. First, we apply QR decomposition to the matrix of cluster centroids $\boldsymbol{C}' = [c_0, \dots, c_K, o]$ to generate the ($\boldsymbol{K}$+1)-dimensional cluster-level ETF classifier. For the class-level ETF classifier, we create an ETF structure for each cluster individually using the prototype matrix of the classes within that cluster, resulting in $\boldsymbol{K}$ distinct class-level ETFs. Since there are no subclasses assigned to the OOD cluster at this stage, the class-level ETFs are defined only for the $\boldsymbol{K}$ clusters obtained from prototype clustering.

### 4.1.2. Incremental Sessions

**Out-of-Distribution Cluster Assignment.** When new classes arrive during incremental sessions, we first extract new class prototypes matrix $\boldsymbol{P}_t$ and determine whether each class belongs to the in-distribution (ID) or the out-of-distribution (OOD) set. This step ensures that classes distributions significantly different from the base classes are assigned to the OOD cluster before the remaining classes are assigned to the $\boldsymbol{K}$ ID clusters obtained through prototype clustering. For OOD detection, we adopt the *neural collapse-based OOD detection* (*NECO*) [40] technique. The *NECO* score is

computed as the relative norm of a new class prototype within the principal space spanned by the base prototypes, normalized by the norm of the new class prototype itself. Given principal space $P \in \mathbb{R}^{D\times d}$ spanned by the $d$ eigenvectors of the latent space $H \in \mathbb{R}^{D\times d}$, the *NECO* score can be formally described as:

$$NECO(x) = \frac{\|Ph_\omega(x)\|}{\|h_\omega(x)\|} = \frac{\sqrt{h_\omega(x)^\top PP^\top h_\omega(x)}}{\sqrt{h_\omega(x)^\top h_\omega(x)}}$$

where $h_\omega(x)$ is the feature vector of a certain input image $x$. If the *NECO* score falls below a pre-defined threshold *r*, the class is categorized as OOD. The *NECO* score ranges from 0 to 1. Classes identified as OOD are then assigned to the OOD cluster as its subclasses.

**In-Distribution Cluster Assignment.** New classes identified as in-distribution (ID) are assigned to the clusters formed during the base session based on cosine similarity. Specifically, we compute the cosine similarity between each new class prototype and all $\boldsymbol{K}$ ID cluster centroids $\boldsymbol{C} = [c_0, \dots, c_K]$, and assign each class to the cluster corresponding to the most similar centroid.

**Hierarchical ETF Update.** The hierarchy is updated based on the cluster assignments of the newly introduced classes, and the class-level ETF classifier is recalculated accordingly. For clusters with newly added classes, we recompute the class-level ETF by performing QR decomposition on the updated prototype matrix of each cluster and deriving the orthogonal matrix $\boldsymbol{Q}$. The cluster-level ETF classifier remains fixed, which allows the model to retain the existing hierarchy information and effectively utilize it in later sessions.

## 4.2. Dynamic Hierarchical Feature-Classifier Alignment

This section describes how the hierarchical ETF classifier constructed in Section 4.1. is utilized for incremental training. During training, the base encoder is kept frozen, and only the projection layers appended to the encoder are updated. The projection layers are organized in a dual-level structure, consisting of a cluster-level projection layer and a class-level projection layer. Each layer is trained using the proposed Hierarchical Dot-Regression (HDR) loss, which encourages the feature representations at each level to align closely with their respective target ETF vectors. The HDR loss is further regularized by the dependency loss, which enforces predictions to be consistent with the hierarchical structure. Following a commonly adopted strategy in FSCIL studies [17, 21], we include base class prototypes $\boldsymbol{P}_0$ for training in incremental sessions to reduce catastrophic forgetting.

### 4.2.1. Hierarchical Dot-Regression Loss

To train the dual-level projection layers, we propose the Hierarchical Dot-Regression (HDR) loss, which aligns the feature representations at each level to the hierarchical ETF classifiers. The HDR loss consists of two dot-regression (DR) loss terms—one for each level of the hierarchy— and an additional dependency loss (described in Section 4.2.2.). The DR loss is derived by simplifying the conventional cross-entropy (CE) loss to eliminate the "push" term [34]. The gradient of CE loss is composed of a pull term and a push term. The pull term attracts features towards their classifier prototype of the same class. The push term repulses the features away from the prototypes of other classes. The DR loss retains only the "pull" component. This design assumes that the optimal class prototypes are already provided by the ETF classifier, making the push term redundant. Formally, the DR loss for a normalized feature $\hat{\mu}_i$ and its corresponding ETF prototype $e_{y_i}$ is given by:

$$\mathcal{L}(\hat{\mu}_i, E) = \frac{1}{2}\left(e_{y_i}{}^{\top}\hat{\mu}_i - 1\right)^2$$

where $E$ is the ETF classifier matrix and $e_{y_i}$ is the prototype of class $y_i$ in $E$. By focusing solely on the "pull" gradients, the DR loss effectively aligns the feature representations with the optimal ETF structure without introducing unnecessary repulsion.

From a dual-level perspective, the cluster-level DR loss $\mathcal{L}_1$ trains the cluster-level projection layer $g_1$ using the cluster-level ETF classifier $E_1$ to encourage *inter-cluster* separation. Similarly, the class-level DR loss $\mathcal{L}_2$ trains the class-level projector $g_2$ using the class-level ETF classifier to enhance *intra-cluster* separation. Together, these two terms guide the feature representations toward the respective ETF prototypes at each level of the hierarchy, ensuring both coarse-level and fine-grained discriminability. The dependency loss $\mathcal{L}_D$ enforces the model to produce predictions consistent with the hierarchical structure. The total loss combines these components as:

$$\mathcal{L}_{HDR} = \alpha \cdot \mathcal{L}_1 + \beta \cdot \mathcal{L}_2 + \lambda \cdot \mathcal{L}_D$$

where $\alpha$, $\beta$, and $\lambda$ are the hyperparameters controlling the contributions of each term.

### 4.2.2. Dependency Loss

To enforce consistency between the hierarchical predictions at different levels, we employ the dependency loss $\mathcal{L}_D$, which regularizes the model to make predictions that adhere to the hierarchy defined by prototype clustering. It is designed to penalize cases where the predicted class does not

belong to the predicted cluster, ensuring that the class-level prediction remains consistent with the higher-level cluster assignment. By explicitly incorporating this hierarchical dependency into the training objective, the model learns to maintain the semantic structure of classes during inference. Formally, the dependency loss is defined as:

$$\mathcal{L}_D = {\rho_1}^{\mathbb{D}\cdot\mathbb{P}_1}{\rho_2}^{\mathbb{D}\cdot\mathbb{P}_2} - 1$$

where $\rho_i$ is a constant penalty factor for level $i$. The indicator terms $\mathbb{D}$ and $\mathbb{P}_i$ are defined as:

$$\mathbb{D} = \begin{cases} 1 & \text{if } \hat{y}_2 \nRightarrow \hat{y}_1 \\ 0 & \text{if } \hat{y}_2 \Rightarrow \hat{y}_1 \end{cases} \qquad \mathbb{P}_i = \begin{cases} 1 & \text{if } \hat{y}_i \neq y_i \\ 0 & \text{if } \hat{y}_i = y_i \end{cases}$$

where $\hat{y}_i$ is the predicted class at level $i$, and $y_i$ is the corresponding ground-truth label. Here, $\mathbb{D}$ activates the penalty if the class-level prediction does not fall within the predicted cluster, and $\mathbb{P}_i$ indicates misclassification at level $i$. Through this design, the dependency loss enforces the model to respect the hierarchy, improving consistency in the hierarchical predictions.

# 5. Experiments

In this section, we evaluate our method on the CIFAR-100 [23] dataset and compare the results with state-of-the-art FSCIL methods. We also visualize the representation space using UMAP [41] projection to assess the hierarchical feature-classifier alignment. Finally, we conduct an evaluation to verify the effectiveness of base prototype clustering and OOD detection.

## 5.1. Experimental Settings

**Dataset.** Following previous works in FSCIL, we conduct experiments on CIFAR-100 benchmark dataset [23] using the data splits in [11]. 60 classes are allocated for base session with 500 training samples per class. Each of the 8 incremental sessions contains 5 classes with 5 training samples per class (*i.e.,* a 5-way 5-shot setting).

**Evaluation Metrics.** Following the popular evaluation protocol of FSCIL [11, 16], we adopt the average accuracy (*aAcc*) as the primary evaluation metric. Unlike prior works that report only the overall average accuracy, we additionally report the average accuracy on base classes ($aAcc_b$) and

incremental classes ($aAcc_i$) separately. Furthermore, we propose the base-incremental gap (BIG) as a new metric to quantify the performance disparity between base and incremental classes. BIG is designed to measure the degree of base-class bias in the model, computed as the average difference between base and incremental performance across sessions. Therefore, lower BIG values indicate better balance. However, since it captures only the raw performance gap, it should be interpreted in conjunction with the absolute performance of base and incremental classes.

**Compared Methods.** We compare our results against three baselines: TEEN [42], LIMIT [20], and FACT [18]. TEEN calibrates new class prototypes in the embedding space based on their semantic similarity to the base classes. This idea makes it a relevant baseline, as it also leverages inter-class semantic relationships, sharing the same intuition as our approach. LIMIT and FACT are forward compatible methods designed to prepare for future updates during the base session.

**Implementation Details.** All methods are implemented with PyTorch [43]. We use ResNet-12 [3] as our backbone network, whereas the baseline methods use ResNet-20. Our model is trained with a batch size of 256, and an initial learning rate of 0.25 and 0.05 with cosine annealing decay for the base session and incremental sessions, respectively. We use SGD with momentum for optimization. All baseline methods are reproduced in our experimental environment using the hyperparameters reported in their original papers.

## 5.2. Results

### 5.2.1. Performance Comparison with Other FSCIL Methods

Our experimental results on CIFAR-100 are presented in Table 1. The results demonstrate that our method outperforms other approaches in per-session performance and overall average accuracy. Specifically, the overall average accuracy of our method is 1.57 percentage points higher than that of the second-best method FACT [18]. For base class accuracy ($aAcc_b$), our method achieves the highest performance in all sessions except **S1**, and for incremental class accuracy ($aAcc_i$), it yields the highest in all sessions except **S4**. Notably, in sessions **S1**, **S2**, and **S3**, our method outperforms the second-best method in $aAcc_i$ by substantial margins of 15.7, 5.0, and 6.2 percentage points, respectively. On average, our method achieves the highest base and incremental class accuracies among all methods. Especially, the incremental performance is improved by nearly 4 percentage points compared to others. Furthermore, BIG is much lower at 46.89 compared to other methods. Maintaining higher base and incremental accuracies while achieving a lower BIG indicates that our method mitigates model bias more effectively than existing FSCIL approaches. Appendix A provides

a statistical analysis of these performance differences using McNemar tests [44], further supporting the significance of our results.

| Methods | Accuracy in each session (%) ↑ | | | | | | | | | Avg. ↑ | BID ↓ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Base | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | | |
| **TEEN** [42] | 78.27 | 72.02 | 67.79 | 63.73 | 60.69 | 57.65 | 55.47 | 52.87 | 51.01 | 62.17 | 48.76 |
| Base class acc. | 78.27 | 75.50 | 74.65 | 73.87 | 73.33 | 72.42 | 71.52 | 71.28 | 70.67 | 73.50 | |
| Incremental class acc. | - | 30.30 | 26.60 | 23.20 | 22.75 | 22.20 | 23.37 | 21.29 | 21.53 | 23.89 | |
| **LIMIT** [20] | 76.15 | 71.71 | 67.36 | 63.19 | 60.08 | 57.24 | 55.12 | 53.08 | 50.80 | 61.64 | 51.22 |
| Base class acc. | 76.15 | 75.42 | 74.48 | 73.65 | 73.10 | 72.43 | 72.18 | 71.98 | 71.25 | 73.40 | |
| Incremental class acc. | - | 27.20 | 24.60 | 21.33 | 21.00 | 20.76 | 21.00 | 20.69 | 20.13 | 22.09 | |
| **FACT** [18] | 78.32 | 72.05 | 68.19 | 63.91 | 60.71 | 57.78 | 55.63 | 53.35 | 51.54 | 62.39 | 49.59 |
| Base class acc. | 78.32 | 75.65 | 75.05 | 74.20 | 73.57 | 72.53 | 71.82 | 71.60 | 70.98 | 73.75 | |
| Incremental class acc. | - | 28.80 | 27.00 | 22.73 | 22.15 | 22.36 | 23.27 | 22.06 | 22.37 | 23.84 | |
| **Ours** | **81.68** | **72.85** | **70.44** | **66.21** | **61.83** | **58.75** | **57.38** | **54.69** | **51.83** | **63.96** | **46.89** |
| Base class acc. | 81.68 | 75.08 | 76.85 | 75.42 | 75.30 | 73.75 | 74.38 | 73.30 | 70.52 | 75.14 | |
| Incremental class acc. | - | 46.00 | 32.00 | 29.40 | 21.40 | 22.76 | 23.37 | 22.80 | 23.80 | 27.69 | |

**Table 1.** Comparison of FSCIL methods across multiple sessions on CIFAR-100. **S0** represents the base session and **S1-S8** denote incremental sessions. The best results are **bolded**.

### 5.2.2. Visualization of Feature Space

Figures 3-5 visualize the learned representations of our method using UMAP [41] projection to map the cluster-level and class-level features into a 2D space. Figure 3 illustrates the results when visualizing only the ID classes. Figure 4 shows the feature space when visualizing both ID and OOD classes. For visualization, we selected eight classes in total—two classes from each of four different clusters—comprising four base classes and four incremental classes, with five samples per class. Within this criterion, all samples and class selections were performed randomly. For the OOD classes, two classes were randomly selected and five samples per class were randomly drawn. Additionally, we compare the feature space of our method with that of TEEN [42], one of the baseline methods, and the results are presented in Figure 5.

**ID Classes.** Figure 3a shows the cluster-level features of the ID classes. We can observe that the features are well grouped according to their respective clusters, and when colored by class, the two classes belonging to the same cluster are grouped together. Figure 3b shows the class-level features

of the ID classes. Compared to the cluster-level features, the classes that were grouped within the same cluster are now more distinctly separated at the class level.

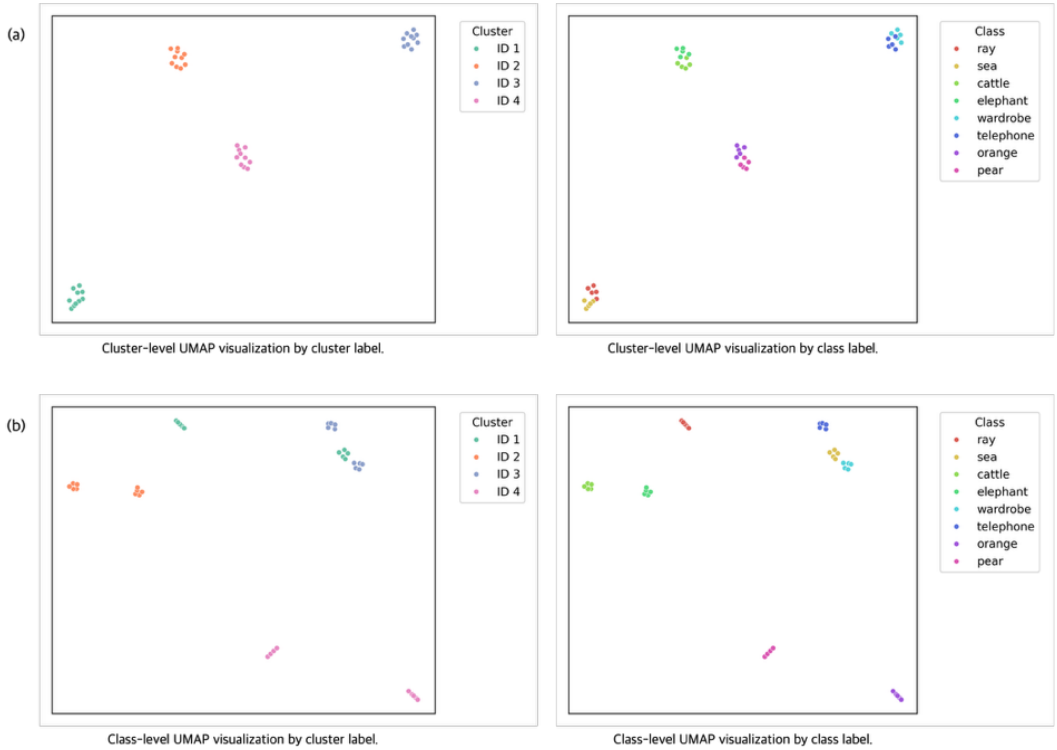



**Figure 3.** UMAP visualization of ID feature space. (left) colored by cluster label. (right) colored by class label. (a) represents the cluster-level feature space, and (b) represents the class-level feature space.

**ID and OOD Classes.** Figure 4 shows the ID classes from Figure 3 with two additional randomly selected OOD classes. Classes “rose” and “wolf” are the OOD classes. In Figure 4a, the cluster-level features remain well-separated by cluster, with classes in the same cluster grouped closely together. However, the OOD cluster does not appear distinctly separated from the others. This is likely due to the nature of the CIFAR-100 [23] dataset, where the distributions of all classes are relatively similar. In Figure 4b, the class-level features demonstrate that features belonging to the same class are well-clustered together.

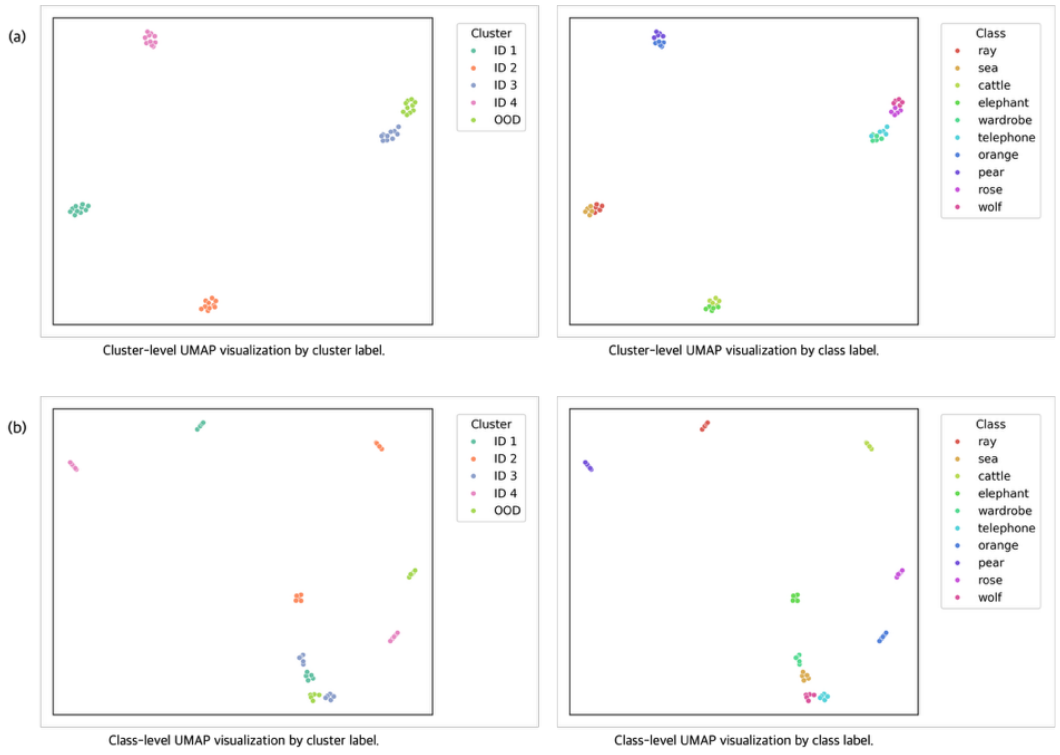
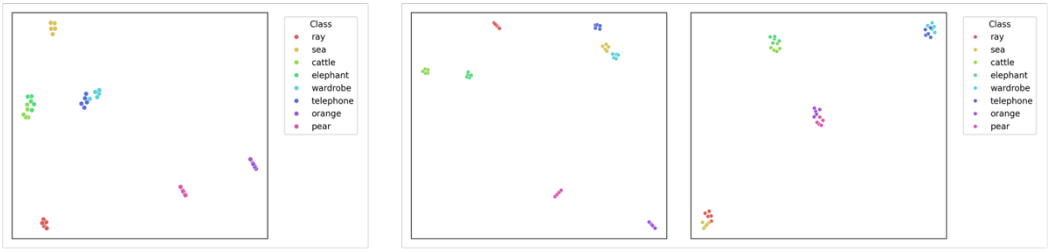



**Figure 4.** UMAP visualization of ID and OOD feature space. (left) colored by cluster label, (right) colored by class label. (a) represents the cluster-level feature space, and (b) represents the class-level feature space.

**Comparison with TEEN.** Figure 5 compares the class-level features of our proposed method and TEEN [42]. As shown in Figure 5a, the TEEN results exhibit good overall class separation, but classes “cattle” and “elephant”, as well as classes “telephone” and “wardrobe”, are distributed close to each other in the feature space. In contrast, Figure 5b shows the results of our method, where these class pairs are more distinctly separated. Notably, when colored by cluster, classes “cattle” and “elephant”, and classes “telephone” and “wardrobe”, each belong to the same cluster. This result demonstrates that our hierarchy-based dual-level training method effectively separates semantically similar classes that are hard to distinguish.




(a) UMAP visualization of TEEN [42] feature space by class label.

(b) UMAP visualization of our feature space by class label.
(Left) Class-level visualization. (Right) Cluster-level visualization.

**Figure 5.** Class-level feature space comparison with TEEN. (a) represents the feature space of TEEN, and (b) represents the feature space of our method. (left) class-level visualization, (right) cluster-level visualization.

### 5.2.3. Evaluation of Base Prototype Clustering

To assess the effectiveness of the base prototype clustering, which serves as the foundation for our hierarchical label construction, we conducted both qualitative and quantitative evaluations with class names and Rand index [45]. Table 2 presents the clustering results of base prototypes mapped to the CIFAR-100 class names. We observe that semantically similar classes are generally grouped into the same cluster, indicating that the clustering aligns well with human intuition. Furthermore, the Rand index computed with the CIFAR-100 coarse labels achieves a high value of 0.9446. This shows strong consistency between our clustering and the dataset's inherent hierarchical structure. These results demonstrate that the constructed hierarchy captures meaningful semantic relationships among classes, providing a reliable basis for the subsequent hierarchical training.

| Cluster ID | Class Names |
|---|---|
| 1 | clock |
| 2 | aquarium fish, crab, crocodile, dinosaur, flatfish, lizard, lobster, mushroom |
| 3 | forest, maple tree, mountain, oak tree, palm tree, pine tree |
| 4 | bottle, can, cup, lamp |
| 5 | baby, boy, girl, man |
| 6 | cloud, dolphin |
| 7 | bee, beetle, butterfly, caterpillar, cockroach, orchid |
| 8 | bicycle, lawn mower, motorcycle |
| 9 | fox, hamster, kangaroo, leopard, lion |
| 10 | apple, bowl, orange, pear |
| 11 | bridge, castle, house |
| 12 | bed, chair, couch, keyboard |
| 13 | cattle, elephant |
| 14 | bus, pickup truck |
| 15 | bear, beaver, camel, chimpanzee, mouse, otter |

**Table 2.** Base prototype clustering results mapped with CIFAR-100 class names.

### 5.2.4. Evaluation of Out-of-Distribution Detection

In this section, we evaluate the effectiveness of *neural collapse-based OOD detection* (*NECO*) [40] on the incremental classes. Figure 6 presents the ID and OOD class distributions in the last session using a kernel density estimation (KDE) [46] plot of the PCA-1D projections of incremental class prototypes. The incremental set consists of 40 classes in total, including 32 ID classes and 8 OOD classes. The KDE plot shows a noticeable separation between the two distributions, suggesting that the ID and OOD classes are well distinguished during the incremental sessions. To further validate the clustering results after the final incremental session, we compute the Rand index [45] over all 100 classes. The overall Rand index achieved a value of 0.8962, showing that the OOD detection and prototype clustering maintained decent alignment with the underlying class semantics even with the increasing number of classes. Additional qualitative analysis of OOD feature distributions over incremental sessions can be found in Appendix B.

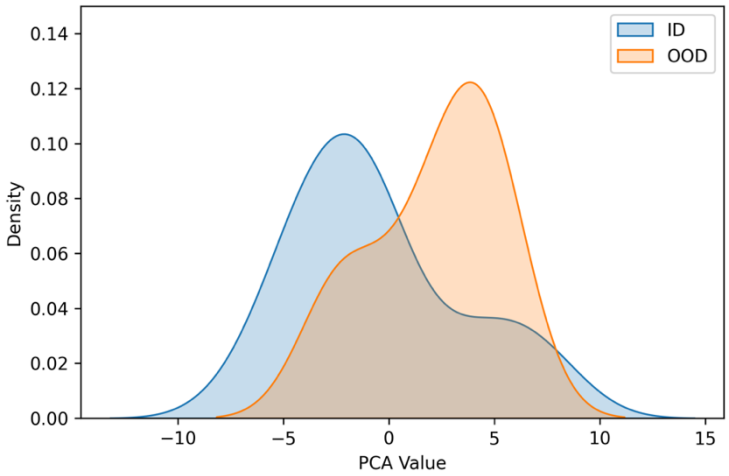



**Figure 6.** Kernel density estimation (KDE) plot of ID and OOD prototypes. The visualization is based on the PCA-1D projections of prototypes. Both ID and OOD prototypes are all from incremental classes.

# 6. Discussion

Few-shot class-incremental learning (FSCIL) requires a model to continuously adapt to newly introduced classes while preserving the knowledge of previously learned ones, which is essential in many real-world applications [8-10]. The FSCIL task is composed of a base session that trains the backbone with sufficient training data including a large number of classes, and multiple incremental sessions where new classes arrive with only a few training samples per class [13]. Such severe class imbalance between base and incremental classes diminishes the model's ability to discriminate new classes and biases its performance toward the base classes.

To address this bias and enhance the model's adaptability to incremental classes, recent works have adopted *forward compatible* techniques [18-21] that aim to reserve space for future classes by learning compact base representations. Among these methods, NC-FSCIL [21] introduces the idea of neural collapse to FSCIL [22], and achieves state-of-the-art performance by pre-defining the ETF classifier during the base session and aligning the new class features to this fixed classifier. However, this method exhibits a critical limitation in class-incremental scenarios. As shown in Figure 1, NC-FSCIL [21] consistently underperforms when the ETF classifier dimension is either smaller or larger than the total number of classes in the dataset, compared to when it matches exactly. This indicates

that constructing an ideal feature space using the ETF structure requires prior knowledge of the total number of classes, which is unrealistic in FSCIL where new classes continue to emerge over time.

In this study, we propose a novel Dynamic Hierarchical ETF (DH-ETF) classifier to tackle this limitation. Instead of relying on a single, fixed set of ETF prototypes pre-defined in the base session, DH-ETF decouples the classifier into two hierarchical levels, each with a distinct role. The "fixed" cluster-level ETF classifier retains a stable global semantic structure that captures the relationships among classes. In contrast, the "adaptive" class-level ETF classifier is dynamically updated within each cluster as new classes arrive, allowing the model to flexibly accommodate the growing number of classes. Using the cluster-level ETF as an anchor for preserving previously acquired knowledge, this hierarchical design mitigates the representation shift caused by updating the class-level ETFs. The effectiveness of our design is supported by the results in Figure 1. As depicted in the figure, our method consistently outperforms NC-FSCIL across all ETF dimensions, except when the dimension is set to 100—a setting that implicitly assumes prior knowledge of the total number of classes. This result empirically validates that our hierarchical and dynamic approach is better suited to the open-ended nature of incremental learning tasks, where the number of classes is unknown and continues to grow incrementally.

Furthermore, to address the base class bias prevalent in existing FSCIL methods, we leverage the semantic similarity among classes to guide the incremental process by ensuring that the newly introduced classes are aligned with semantically relevant clusters. This use of class semantics not only enhances inter-cluster separability through cluster-level optimization, but also facilitates intra-cluster separation by enabling fine-grained differentiation of semantically similar few-shot classes within each cluster. As a result, our method improves the model's ability to effectively discriminate incremental classes, thereby alleviating the bias toward base classes.

Experimental results on the CIFAR-100 [23] benchmark dataset demonstrate the effectiveness of our approach. As shown in Table 1, it outperforms state-of-the-art FSCIL methods in both base and incremental class accuracy and achieves the lowest base-incremental performance gap (BIG). Maintaining higher base and incremental class performance while reducing the BIG suggests that our method mitigates the base class bias more effectively than existing FSCIL approaches. Feature space visualizations in Figures 3-5 further indicate that our method produces semantically consistent feature distributions, reflecting the intended hierarchical structure. In Figures 3 and 4, the cluster-level features are well separated by class groups, and the classes grouped within the same cluster at the cluster level are further divided at the class level. This presents that our method ensures both inter-class and intra-class separation through its hierarchical structure.

In addition, Table 2 and Figure 6 highlight the robustness of prototype clustering and out-of-distribution (OOD) detection mechanisms. As shown in Table 2, the clustering results for base class prototypes well aligns with human intuition as semantically similar classes are generally grouped into the same cluster. The kernel density estimation (KDE) plot in Figure 6 presents a noticeable

separation between the ID and OOD incremental class distributions. These results suggest that our method retains meaningful semantic alignment even as new classes are introduced incrementally.

While our method demonstrates promising results, several limitations remain. First, although it achieved high quantitative clustering quality and visually interpretable class clusters, we observed occasional feature overlap between classes from different clusters at finer levels of the hierarchy. This suggests the need for additional constraints to ensure that intra-cluster discriminability does not compromise inter-cluster separation. Second, the performance drop rate between the base session and the last session remains noticeable and requires further improvement. Lastly, our experiments are conducted on a single benchmark dataset with relatively balanced distributions. Further work should validate the effectiveness of our method on datasets with more diverse and challenging distributions, as well as in real-world applications, to strengthen its generalizability.

# 7. Conclusion

In this study, we propose a novel Dynamic Hierarchical ETF (DH-ETF) classifier to address the challenges of few-shot class-incremental learning (FSCIL), including the strong bias toward base classes and impracticality of fixed classifier designs. By decoupling the classifier into a fixed cluster-level and an adaptive class-level structure, and leveraging semantic hierarchy, our method achieves a better balance between stability and adaptability. Experimental results demonstrate that DH-ETF not only outperforms state-of-the-art methods for both base and incremental classes but also retains semantically meaningful feature representations throughout the incremental process. In addition, quantitative and qualitative evaluations of prototype clustering and OOD detection schemes further support the efficacy of our design. These findings highlight the potential of integrating hierarchical semantics into FSCIL to overcome the limitations of existing methods and pave the way for future research in more diverse and challenging real-world scenarios.

# Appendix A. Statistical Significance Test

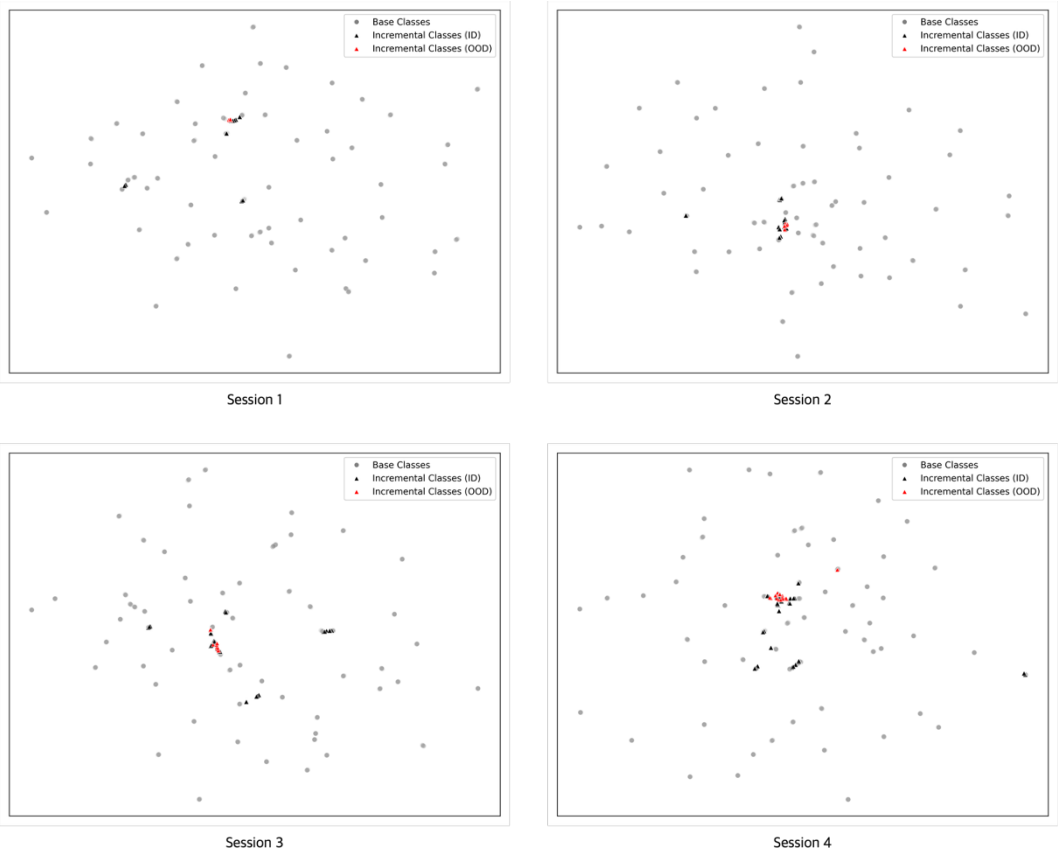
To assess whether the performance differences between our method and the baseline methods are statistically significant, we conducted the McNemar test [44] at each incremental session. The McNemar test compares the discordant pairs between two classifiers by assigning a value of 1 to correctly classified samples and 0 to misclassified samples, and determines whether the observed difference in learning patterns is statistically significant. Table A.1 summarizes the p-values of the McNemar test for each session, comparing our method against TEEN [42], LIMIT [20], and FACT [18]. At a significance level of 0.05, most sessions exhibit statistically significant differences in the error patterns between our method and each baseline. These results implies that the improvements in accuracy achieved by our method is statistically significant compared to the baseline approaches.

| | **McNemar Test p-values in each session (significance level = 0.05)** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Methods** | **S1** | **S2** | **S3** | **S4** | **S5** | **S6** | **S7** | **S8** |
| **vs. TEEN** [42] | **0.0259** | 0.2012 | **0.0259** | 0.0890 | **0.0001** | 0.3268 | **0.0001** | 0.1374 |
| **vs. LIMIT** [20] | **0.0093** | 0.2012 | **0.0062** | **0.0311** | **0.0001** | 0.0543 | **0.0001** | **0.0148** |
| **vs. FACT** [18] | **0.0008** | **0.0209** | **0.0013** | **0.0071** | **0.0001** | 0.1356 | **0.0001** | 0.0543 |

**Table A.1.** McNemar test p-values in each incremental session. The test is conducted to statistically compare the performance of our method and each baseline method. Statistically significant results at the significance level of 0.05 are **bolded**.

# Appendix B. Visualization of Feature Distributions Across Sessions

To further analyze the behavior of our model, we visualize the feature distributions of all classes in each incremental session using UMAP [41]. Visualizations include base classes, and incremental classes identified as in-distribution (ID) and out-of-distribution (OOD). As shown in Figure B.1, the proportion of classes detected as OOD remains relatively low across all sessions. Given the large size and diversity of the base class set, the majority of incremental classes are identified as ID and appropriately assigned to semantically relevant clusters. Additionally, the OOD incremental classes (red triangles) consistently form a compact cluster near the origin in the feature space, rather than being scattered. This indicates that even with just one dedicated OOD cluster, our model is able to maintain appropriate class separation, while preserving meaningful structure within the ID space.



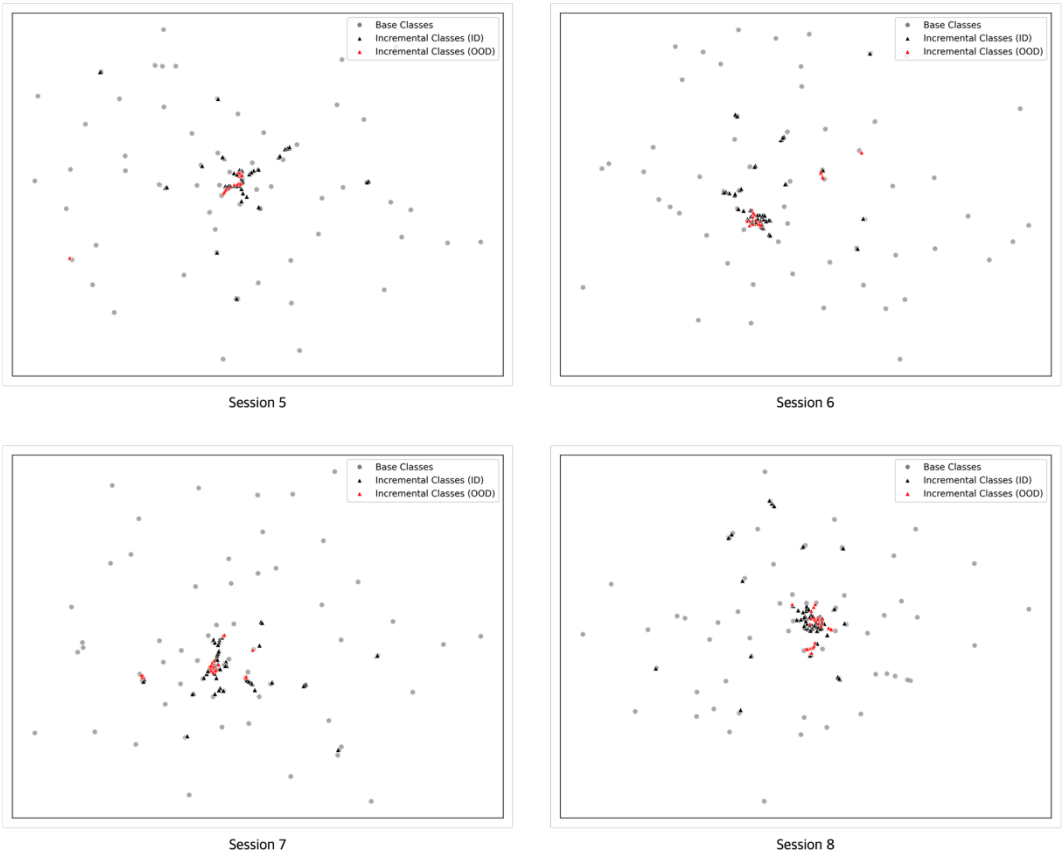



**Figure B.1.** UMAP visualization of feature distributions across sessions. Gray circles represent base classes, black triangles indicate in-distribution (ID) incremental classes, and red triangles represent out-of-distribution (OOD) incremental classes.

# References

[1] B. K. Jang *et al.*, "Classification models for arthropathy grades of multiple joints based on hierarchical continual learning," (in English), *Radiol Med,* vol. 130, no. 6, pp. 782-794, Jun 2025, doi: 10.1007/s11547-025-01974-4.

[2] J. Jumper *et al.*, "Highly accurate protein structure prediction with AlphaFold," (in English), *Nature,* vol. 596, no. 7873, pp. 583-+, Aug 26 2021, doi: 10.1038/s41586-021-03819-2.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[4] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences,* vol. 3, no. 4, pp. 128-135, 1999.

[5] G. M. Van de Ven, T. Tuytelaars, and A. S. Tolias, "Three types of incremental learning," *Nature Machine Intelligence,* vol. 4, no. 12, pp. 1185-1197, 2022.

[6] S. Yan, J. Xie, and X. He, "Der: Dynamically expandable representation for class incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 3014-3023.

[7] D.-W. Zhou, Q.-W. Wang, Z.-H. Qi, H.-J. Ye, D.-C. Zhan, and Z. Liu, "Class-incremental learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2024.

[8] M. Sirshar, T. Hassan, M. U. Akram, and S. A. Khan, "An incremental learning approach to automatically recognize pulmonary diseases from the multi-vendor chest radiographs," *Computers in Biology and Medicine,* vol. 134, p. 104435, 2021.

[9] L. Sun, M. Zhang, B. Wang, and P. Tiwari, "Few-shot class-incremental learning for medical time series classification," *IEEE Journal of Biomedical and Health Informatics,* vol. 28, no. 4, pp. 1872-1882, 2023.

[10] J. M. Pierre, "Incremental lifelong deep learning for autonomous vehicles," in *2018 21st international conference on intelligent transportation systems (ITSC)*, 2018: IEEE, pp. 3949-3954.

[11] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12183-12192.

[12] J. Zhang, L. Liu, O. Silvén, M. Pietikäinen, and D. Hu, "Few-Shot Class-Incremental Learning for Classification and Object Detection: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2025.

[13] S. Tian, L. Li, W. Li, H. Ran, X. Ning, and P. Tiwari, "A survey on few-shot class-incremental learning," *Neural Networks,* vol. 169, pp. 307-324, 2024.

[14] Y. Shen, Y. Xiong, W. Xia, and S. Soatto, "Towards backward-compatible representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6368-6377.

[15] S. Dong, X. Hong, X. Tao, X. Chang, X. Wei, and Y. Gong, "Few-shot class-incremental learning via relation knowledge distillation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, no. 2, pp. 1255-1263.

[16] C. Zhang, N. Song, G. Lin, Y. Zheng, P. Pan, and Y. Xu, "Few-shot incremental learning with continually evolved classifiers," in *Proceedings of the IEEE/CVF conference on*

*computer vision and pattern recognition*, 2021, pp. 12455-12464.

[17] M. Hersche, G. Karunaratne, G. Cherubini, L. Benini, A. Sebastian, and A. Rahimi, "Constrained few-shot class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 9057-9067.

[18] D.-W. Zhou, F.-Y. Wang, H.-J. Ye, L. Ma, S. Pu, and D.-C. Zhan, "Forward compatible few-shot class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 9046-9056.

[19] C. Peng, K. Zhao, T. Wang, M. Li, and B. C. Lovell, "Few-shot class-incremental learning from an open-set perspective," in *European Conference on Computer Vision*, 2022: Springer, pp. 382-397.

[20] D.-W. Zhou, H.-J. Ye, L. Ma, D. Xie, S. Pu, and D.-C. Zhan, "Few-shot class-incremental learning by sampling multi-phase tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 45, no. 11, pp. 12816-12831, 2022.

[21] Y. Yang, H. Yuan, X. Li, Z. Lin, P. Torr, and D. Tao, "Neural collapse inspired feature-classifier alignment for few-shot class incremental learning," *arXiv preprint arXiv:2302.03004,* 2023.

[22] V. Papyan, X. Han, and D. L. Donoho, "Prevalence of neural collapse during the terminal phase of deep learning training," *Proceedings of the National Academy of Sciences,* vol. 117, no. 40, pp. 24652-24663, 2020.

[23] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[24] Y. He, Y. Chen, Y. Jin, S. Dong, X. Wei, and Y. Gong, "Dyson: Dynamic feature space self-organization for online task-free class incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23741-23751.

[25] G. An, M. Akiba, K. Omodaka, T. Nakazawa, and H. Yokota, "Hierarchical deep learning models using transfer learning for disease detection and classification based on small number of medical images," *Scientific reports,* vol. 11, no. 1, p. 4250, 2021.

[26] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski, "Hierarchical annotation of medical images," *Pattern Recognition,* vol. 44, no. 10-11, pp. 2436-2449, 2011.

[27] K. Kowsari *et al.*, "Hmic: Hierarchical medical image classification, a deep learning approach," *Information,* vol. 11, no. 6, p. 318, 2020.

[28] L. Zhu *et al.*, "Hcl4qc: Incorporating hierarchical category structures into contrastive learning for e-commerce query classification," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023, pp. 3647-3656.

[29] H. Cotacallapa, N. Saboya, P. C. Rodrigues, R. Salas, and J. L. López-Gonzales, "A flat-hierarchical approach based on machine learning model for e-commerce product classification," *IEEE Access,* 2024.

[30] M. Bader-El-Den, E. Teitei, and M. Adda, "Hierarchical classification for dealing with the Class imbalance problem," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016: IEEE, pp. 3584-3591.

[31] R. M. Pereira, Y. M. Costa, and C. N. Silla Jr, "Toward hierarchical classification of imbalanced data using random resampling algorithms," *Information Sciences,* vol. 578, pp. 344-363, 2021.

[32] H. Xiong and A. Yao, "Deep imbalanced regression via hierarchical classification adjustment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

*Recognition*, 2024, pp. 23721-23730.

[33] B. H. Lee, O. Jung, J. Choi, and S. Y. Chun, "Online continual learning on hierarchical label expansion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 11761-11770.

[34] Y. Yang, S. Chen, X. Li, L. Xie, Z. Lin, and D. Tao, "Inducing neural collapse in imbalanced learning: Do we really need a learnable classifier at the end of deep neural network?," *Advances in neural information processing systems,* vol. 35, pp. 37991-38002, 2022.

[35] L. Xie, Y. Yang, D. Cai, and X. He, "Neural collapse inspired attraction–repulsion-balanced loss for imbalanced learning," *Neurocomputing,* vol. 527, pp. 60-70, 2023.

[36] C. Thrampoulidis, G. R. Kini, V. Vakilian, and T. Behnia, "Imbalance trouble: Revisiting neural-collapse geometry," *Advances in Neural Information Processing Systems,* vol. 35, pp. 27225-27238, 2022.

[37] T. Galanti, A. György, and M. Hutter, "On the role of neural collapse in transfer learning," *arXiv preprint arXiv:2112.15121,* 2021.

[38] L. Xiao *et al.*, "Principled and efficient transfer learning of deep models via neural collapse," *Transactions on Machine Learning Research,* 2024.

[39] Q. Wang *et al.*, "Dualcp: Rehearsal-free domain-incremental learning via dual-level concept prototype," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025, vol. 39, no. 20, pp. 21198-21206.

[40] M. B. Ammar, N. Belkhir, S. Popescu, A. Manzanera, and G. Franchi, "Neco: Neural collapse based out-of-distribution detection," *arXiv preprint arXiv:2310.06823,* 2023.

[41] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426,* 2018.

[42] Q.-W. Wang, D.-W. Zhou, Y.-K. Zhang, D.-C. Zhan, and H.-J. Ye, "Few-shot class-incremental learning via training-free prototype calibration," *Advances in Neural Information Processing Systems,* vol. 36, pp. 15060-15076, 2023.

[43] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems,* vol. 32, 2019.

[44] M. Q. Pembury Smith and G. D. Ruxton, "Effective use of the McNemar test," *Behavioral Ecology and Sociobiology,* vol. 74, no. 11, p. 133, 2020.

[45] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical association,* vol. 66, no. 336, pp. 846-850, 1971.

[46] S. Węglarczyk, "Kernel density estimation and its application," in *ITM web of conferences*, 2018, vol. 23: EDP Sciences, p. 00037.

Abstract in Korean

# 동적 특징-분류기 정렬을 통한 소수 샘플 기반 클래스 증분 학습의 편향 완화

소수 샘플 기반 클래스 증분 학습(few-shot class-incremental learning, FSCIL)은 이전에 학습한 지식을 유지하면서, 소수의 학습 샘플만으로 새로운 클래스를 지속적으로 학습해야 하는 문제를 다룬다. 이러한 상황에서는 기존 클래스와 새로운 클래스 간의 심한 불균형으로 인해 모델이 기존 클래스에 편향되고, 새로운 클래스를 효과적으로 받아들이지 못하는 문제가 발생한다. 최근 연구들은 이를 해결하고자 신경망 훈련의 말기에 나타나는 신경 붕괴(neural collapse) 현상의 결과물인 등각 균형 프레임(equiangular tight frame, ETF)의 기하학적 특성을 활용한다. 그러나, 기존의 ETF 기반 방법론은 base 단계에서 전체 클래스의 수를 가지고 미리 정의된 고정 분류기를 사용하는데, 이는 새로운 클래스가 계속해서 추가되는 FSCIL 의 특성에 부합하지 않는다. 본 연구에서는 클래스 간 의미론적 유사성을 활용하여 확장성과 예측력을 모두 확보하는 동적 계층 ETF 분류기를 제안한다. Base 단계 학습 후 base 클래스들의 클래스별 프로토타입을 군집화 하여 계층을 형성하고, 이를 기반으로 고정된 군집 레벨 ETF 와 동적으로 조정 가능한 클래스 레벨 ETF 로 구성된 계층적 ETF 분류기를 구축한다. 각 증분 세션에서는 새로운 클래스의 수와 이들의 base 클래스와의 특징 유사도에 따라 클래스 레벨 ETF 를 업데이트한다. 이러한 구조는 고정된 군집 레벨 표현을 통해 기존의 지식을 보존하면서도, 클래스 레벨 ETF 의 동적인 조정을 통해 새로운 클래스에 유연하게 대처하도록 한다. CIFAR-100 데이터셋에 대한 실험 결과, 제안한 방법은 기존의 최신 방법들보다 우수한 성능을 보이며, 기존 클래스에 대한 치명적 망각(catastrophic forgetting)을 줄이는 동시에 새로운 클래스의 성능을 향상시킴으로써 FSCIL 모델의 기존 클래스 편향 현상을 완화한다.

---

**핵심 되는 말** : 소수 샘플 기반 클래스 증분 학습, 신경 붕괴, 등각 균형 프레임