SOGANG UNIVERSITY

MASTERS THESIS

Neural Collapse-Driven, Uncertainty-Aware Framework for Few-Shot Class-Incremental Learning

Author:
Sungwon WOO

Supervisor:
Jongho NANG

Multimedia System Laboratory
Department of Artificial Intelligence

Contents

List of Figures

List of Tables

Abstract

Few-Shot Class-Incremental Learning (FSCIL) aims to learn new classes from limited samples while preserving knowledge of base classes. However, most existing FSCIL methods suffer from performance imbalance: accuracy is dominated by base classes, often accompanied by an *asymmetric misclassification problem* in which incremental samples are frequently misclassified as base classes. To address this issue, we propose an uncertainty-aware FSCIL framework that jointly models feature representations and predictive uncertainty. An auxiliary MLP branch is trained alongside the main classifier to estimate an uncertainty score for each feature, enabling the model to predict both class identity and confidence. Inspired by the geometric structure of Neural Collapse, we utilize Equiangular Tight Frame (ETF) vectors over the full class set and partition them into base and incremental groups. We then design an uncertainty scoring function based on their relative alignment with incoming features. For post-hoc calibration at each incremental session, we compute feature-wise uncertainty scores using the designed scoring function. Then we estimate the uncertainty distributions of base and incremental classes using memory prototypes and current session samples, respectively. These distributions are used to compute class-aware uncertainty likelihoods, which re-weight cosine similarity scores for improved decision calibration. Our proposed joint modeling and calibration improve robustness under class imbalance and yield consistent accuracy gains over FSCIL baselines.

Keywords

Deep Learning, Few-Shot learning, Continual Learning, Few-Shot Class-Incremental Learning, Uncertainty-Aware Learning, Neural Collapse

요약

Few-Shot Class-Incremental Learning (FSCIL)은 제한된 샘플로 새로운 클래스를 점진적으로 학습하면서, 기존에 학습된 base 클래스의 지식을 유지하는 것을 목표로 한다. 그러나 기존의 FSCIL 방법들은 base 클래스의 성능이 압도적으로 높은 성능 불균형 문제를 겪고 있으며, 이러한 불균형은 종종 *비대칭적 오분류 현상*을 동반하는데, 이는 incremental 샘플들이 base 클래스로 잘못 분류되는 경향을 의미한다. 이를 해결하기 위해, 본 연구에서는 feature 표현과 예측 불확실성을 공동으로 모델링하는 uncertainty-aware FSCIL 프레임워크를 제안한다. 구체적으로, 보조적인 MLP 브랜치를 메인 분류기와 함께 학습하여 각 feature에 대한 uncertainty score를 추정함으로써, 모델이 클래스 식별과 신뢰도를 함께 예측할 수 있도록 한다. Neural Collapse의 기하학적 구조에서 영감을 받아, 전체 클래스 집합에 대해 Equiangular Tight Frame (ETF) 벡터를 정의하고 이를 base 클래스와 incremental 클래스 그룹으로 나누어 구조화된 feature 공간을 구성한다. 이후, incoming feature와 ETF 벡터 간의 상대적 정렬 정도를 기반으로 uncertainty scoring function을 설계한다. Incremental session마다 수행되는 post-hoc calibration 단계에서는, 설계한 scoring function을 통해 feature-wise uncertainty score를 계산한 후, base 클래스는 memory prototype을, incremental 클래스는 현재 세션의 샘플을 활용하여 클래스별 uncertainty 분포를 추정한다. 이 분포를 기반으로 class-aware uncertainty likelihood를 계산하고, 이를 통해 cosine similarity score를 보정하여 의사결정의 정확도를 향상시킨다. 제안하는 representation-uncertainty 공동 학습과 calibration 기법은 base와 incremental 클래스 간 불균형 문제를 완화하고, 기존의 강력한 FSCIL baseline과 비교하여 세션 전반에 걸쳐 일관된 성능 향상을 달성한다.

주제어

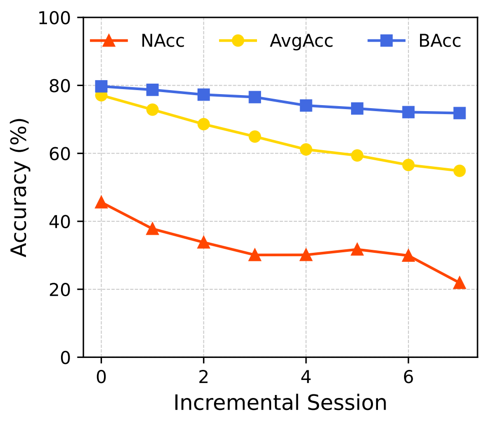
딥러닝, 퓨샷 러닝, 지속 학습, 퓨샷 클래스 증분 학습, 불확실성 인지 학습, 신경망 붕괴 현상

Chapter 1

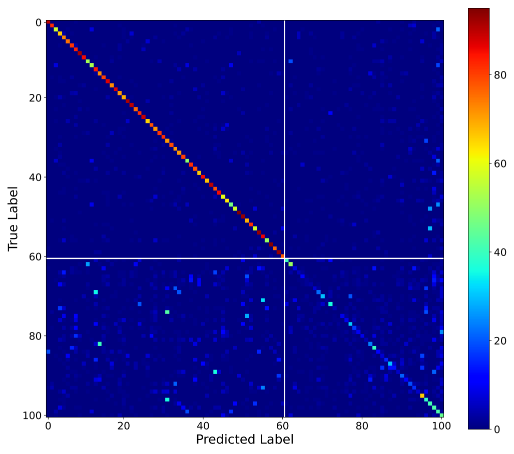
Introduction

1.1 Research background

Deep neural networks (DNNs) have achieved remarkable success in recent years across a wide range of domains [1], [2]. However, many real-world applications require learning from streaming data, where storing all past data is often impractical due to storage limitations or privacy restrictions [3], [4]. In such cases, models must be able to learn new information while retaining previous knowledge, without retraining from scratch. To address this, Class-Incremental Learning (CIL) has been proposed as a learning paradigm where models incrementally learn new classes while preserving knowledge of previously learned ones [5]–[8]. CIL is inherently a dynamic learning problem, and its primary challenge is catastrophic forgetting [9], [10] - the degradation of previous acquired knowledge when adapting to new data. Nevertheless, in many practical scenarios such as medical diagnosis [11] or autonomous driving [12], acquiring a large number of labeled samples for each new class is often infeasible. Instead, only a few labeled examples are available, making conventional CIL methods hard to apply. To overcome this limitation,



(A) Session-wise base, incremental, and average accuracy of CIFAR100. The gap between base and incremental accuracy persists across sessions, leading to average accuracy being disproportionately influenced by the base classes



(B) Confusion matrix at session 9 with CIFAR100. Confusion matrix showing asymmetric misclassification: incremental class samples are frequently misclassified as base classes, while the reverse is rare.

FIGURE 1.1: (a) Accuracy gap across sessions, and (b) confusion pattern at last session.

a more realistic and practical paradigm, Few-Shot Class-Incremental Learning (FSCIL), has been introduced [13]–[20]. Specifically, the FSCIL task consists of a base session with sufficient training data, followed by multiple incremental sessions where an extremely limited number of samples are provided. In addition to catastrophic forgetting, FSCIL introduces a unique challenge of severe class imbalance, which often leads to overfitting on the scarce incremental data. As such, FSCIL requires a careful balance between plasticity for learning novel classes and stability for retaining base knowledge, under both data scarcity and streaming constraints.

Most existing FSCIL methods aim to ensure high stability by adopting an incremental-frozen framework and a prototype-based classifier structure [16], [21]–

[25]. In this setting, the backbone encoder is trained only during the base session using base class data and remains frozen throughout all subsequent incremental sessions. During these incremental sessions, only the classifier head is updated by storing the class-mean feature (i.e. prototypes) extracted from the frozen encoder. While this strategy maintains strong stability, it still suffers from potential overfitting on incremental classes. To mitigate this, recent works have proposed forward-compatible approaches that proactively prepare the feature space during the base session for the inclusion of future classes. Some expand the representational space of base classes [14], [22], [23], others predefine classifier weights for unseen classes [25], or train additional modules to improve generalization to novel classes [24]. However, as shown in Figure 1.1a, a significant performance gap between base and incremental classes remains a persistent issue-base accuracy tends to be high, while incremental accuracy is relatively low. Furthermore, as Figure 1.1b shows, although misclassifying base classes as incremental classes is rare, the opposite is much more common: incremental class samples are frequently misclassified as base classes [21].

In this work, we draw inspiration from the asymmetrical misclassification pattern commonly observed in FSCIL, where incremental class samples are frequently misclassified as base classes, but not vice versa. We hypothesize that this asymmetry reflects an underlying difference in the model's confidence when dealing with base and incremental samples. If the model were able to quantify its uncertainty for each representation, specifically how confident it is about the learned features, it could enable alternative decision strategies, such as deferring high-uncertainty predictions to a human-in-the-loop system [26]. In the FSCIL setting, class imbalance causes the model to be more confident about base classes and less so about incremental classes. Therefore, assigning higher uncertainty scores to incremental samples can serve as a useful signal to identify potentially misclassified instances. Recent studies in classification theory have explored the phenomenon of Neural Collapse, a stage in training where deep representations become highly structured. In this regime, class features collapse to their respective class means, and the classifier weights converge to an Equiangular Tight Frame (ETF) configuration. Inspired by this structure, NC-FSCIL [25] pre-assigns ETF-based classifier weights for both base and future incremental classes. Building upon this idea, we further observe that the degree of alignment between feature vectors and ETF vectors, across both base and incremental classes, varies depending on the sample class type. Specifically, base samples tend to align strongly with base ETF vectors and weakly with incremental ones, while incremental samples are influenced by both to varying degrees. Therefore, we propose to estimate feature-level uncertainty by quantifying how much a given representation is simultaneously aligned with both the base and incremental ETF directions. We argue that this uncertainty score can serve as a signal to identify and correct misclassified incremental samples, thereby mitigating the performance gap between base and incremental classes in FSCIL.

To address the misclassification asymmetry in FSCIL, we propose to leverage the geometric properties of the ETF structure to define an uncertainty score based on the relative alignment between the feature vector and both the base and incremental ETF vectors. Specifically, we compute the maximum cosine similarity between the feature representation and each ETF vector to estimate how strongly the feature is aligned with each class group. Building upon the NC-FSCIL [25], we introduce an uncertainty head that learns to predict an uncertainty score for each feature embedding. The score is designed such that representations more aligned with incremental ETF vectors produce scores near 0, while those closer to base ETF vectors produce scores near 1. As a result, incremental samples are expected to yield uncertainty scores distributed between 0 and 1, whereas base samples produce scores concentrated near 1. To ensure a consistent interpretation of uncertainty across all classes, we apply a transformation of "1 - score" for base samples, so that higher values always indicate higher uncertainty regardless of class type. To the best of our knowledge, this is the first attempt to incorporate uncertainty estimation into the FSCIL framework.

To make use of the uncertainty score during incremental session, we perform a post-hoc calibration procedure at each incremental session. For this, we first compute feature-wise uncertainty score using our scoring function. In practice, we consider two types of uncertainty scores: one predicted by the learned uncertainty head and the other directly derived from the scoring function based on feature-to-ETF alignment. While the predicted uncertainty enables end-to-end learning, it is prone to estimation errors. Therefore, we adopt the scoring-function-based uncertainty for calibration, as it is deterministic and exhibits no deviation by design. Then, we model the uncertainty distributions of base and incremental classes separately—using memory prototypes for base classes and the current session's samples for incremental classes. These class-specific distributions allow us to compute likelihood-based calibration weights, which are used to re-weight the cosine similarity scores between features and classifier weights. This enables the model to adjust its predictions based on how uncertain a sample is with respect to each class group, helping to suppress overconfident predictions on base classes and recover misclassified incremental samples.

The key contributions of this work are summarized as follows:

- We propose a novel uncertainty estimation strategy tailored for FSCIL, leveraging the ETF geometry to quantify how feature representations align with both base and incremental class prototypes.
- We introduce an uncertainty estimation module on top of the NC-FSCIL baseline, allowing the model to output both class predictions and uncertainty scores for each input sample.
- We design a post-hoc calibration method that re-weights prediction scores using class-wise uncertainty distributions estimated from memory prototypes and current session samples, leading to improved classification performance.
- Extensive experiments on multiple benchmarks demonstrate that our method consistently outperforms existing FSCIL baselines across incremental sessions.

Chapter 2

Related Works

2.1 Few-Shot Class-incremental Learning

Few-Shot Class-Incremental Learning (FSCIL) problem focuses on a more practical setting in addition to class-incremental learning, where new sessions are learned from only a few training examples. Due to the constraint of limited data avil-ability, FSCIL presents greater challenges in mitigating catastrophic forgetting and avoiding overfitting on the few-shot incremental data. FSCIL is first introduced by TOPIC [13] which uses neural gas (NG) network to model the topology of fea-ture space. Recent approaches can be broadly categorized into stability-focused and plasticity-focused methods. Stability-focused approaches consolidate the stability by utilizing an incremental-frozen framework, and incorporate various techniques to compensate for limited plasticity. FACT [14] proposes forward compatible train-ing technique to reserve feature spaces for incoming new classes. SAVC [23] incor-porates supervised contrastive loss along with fantasy space to enhance sufficient separation for novel classes. ALICE [22] applies class-level augmentation and data augmentation to improve generalization to the new classes. NC-FSCIL [25] introduces pre-assigned classifer weights (i.e. prototypes) based on a simplex equiangular tight frame (ETF), and fine-tunes a projection layer to align the output features with their corresponding prototypes. Plasticity-focus approaches, on the other hand, enhance the model's ability to learn new classes by fine-tuning the encoder through incremental SubNet tuning [27]. However, none of these methods explicity consider the uncertainty of the feature reprsentations, This motivates the need for uncertainty-aware FSCIL strategies.

2.2 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 loss functions. Recent studies have demonstrated that this geometric collapse offers valuable insights into various learning paradigms, including imbalanced learning [28]–[31], transfer learning [32], [33], and continual learning [25]. To the best of our knowledge, this work is the first to explore uncertainty estimation through the lens of the unique geometric structure induced by Neural Collapse.

2.3 Uncertainty Quantification

Uncertainty quantification (UQ) [34], [35] has become a critical component in modern deep learning systems, particularly in safety-critical applications such as medical diagnosis [36], [37], autonomous driving, and out-of-distribution detection. UQ aims to estimate the degree of uncertainty in a model's prediction, typically by assigning a scalar uncertainty score to each input sample. This score reflects either the model's epistemic uncertainty (stemming from lack of knowledge) or aleatoric uncertainty (arising from inherent data noise). In this work, we apply uncertainty quantification to the FSCIL setting and utilize it for post-hoc calibration, aiming to reduce misclassification of incremental classes.

Chapter 3

Proposed Method

3.1 Preliminary

3.1.1 The FSCIL Setting

In the Few-Shot Class-Incremental Learning (FSCIL) setting, we assume a total of T sessions, consisting of one *base session* (i.e., session 0) and $T-1$ *incremental sessions* (i.e., sessions 1 through $T-1$). The training data for the base session is denoted by $\mathcal{D}_0$, and the training data for the i-th incremental session is denoted as $\mathcal{D}_i = \{(x_j, y_j)\}_{j=1}^{N_i}$, where $\mathcal{C}_i$ is the corresponding label space.

In each session i, only the training data $\mathcal{D}_i$ and label space $\mathcal{C}_i$ are available; data from previous sessions is not accessible. The test set for session i is denoted as $\mathcal{D}_i^{\text{test}}$, and the cumulative test label space is defined as:

$$\mathcal{C}_i^{\text{test}} = \bigcup_{j=0}^{i} \mathcal{C}_j, \tag{3.1}$$

which is used to evaluate the model's discriminative ability over all previously seen classes.

Each incremental session can also be interpreted as an N-way K-shot classification task, where N denotes the number of new classes introduced in the session, and K is the number of labeled examples per class. The label spaces across sessions are disjoint, i.e., for any $i \neq j$, we have $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$.

Compared to standard Class-Incremental Learning (CIL), FSCIL introduces a more challenging scenario by limiting the amount of labeled data in incremental sessions. On the other hand, unlike conventional Few-Shot Learning (FSL), FSCIL requires the model to continually learn new classes while retaining knowledge of all previously seen classes.

The model in FSCIL typically consists of a feature encoder $\phi_\theta(\cdot)$, parameterized by θ, and a linear classifier W. Given an input $x_j \in \mathbb{R}^D$, its feature representation is computed as $\phi_\theta(x_j) \in \mathbb{R}^d$. For a classification task with N classes, the output logits are given by:

$$O_j = W^\top \phi_\theta(x_j) \in \mathbb{R}^N, \quad \text{where } W \in \mathbb{R}^{d \times N}. \tag{3.2}$$

3.1.2 Neural Collapse

Neural Collapse is a phenomenon that arises at the terminal phase of training deep neural networks, particularly after achieving zero training error on balanced datasets. It describes the emergence of a highly symmetric geometric structure involving both the last-layer feature representations and the classifier weights.

Definition 1 (Simplex Equiangular Tight Frame (ETF)) *A set of K vectors $\{e_k\}_{k=1}^K$ in $\mathbb{R}^d$ forms a Simplex Equiangular Tight Frame if the following conditions are satisfied:*

- *All vectors are unit-norm and equiangular, i.e.,*

$$\langle e_{k_1}, e_{k_2} \rangle = \begin{cases} 1, & \text{if } k_1 = k_2, \\ -\frac{1}{K-1}, & \text{if } k_1 \neq k_2, \end{cases}$$

- *The matrix $E = [e_1, e_2, \ldots, e_K] \in \mathbb{R}^{d \times K}$ satisfies $E^\top E = \left(1 + \frac{K}{K-1}\right) I_K - \frac{1}{K-1} \mathbf{1}_K \mathbf{1}_K^\top$, where I_K is the $K \times K$ identity matrix and $\mathbf{1}_K$ is a K-dimensional vector of ones.*

The Neural Collapse phenomenon is characterized by the following four empirical observations:

(NC1) Variance Collapse: For each class k, the last-layer features collapse to their class mean μ_k, and the within-class covariance tends to zero:

$$\Sigma_k^{\text{within}} = \mathbb{E}_{x_i \in \mathcal{C}_k} \left[(\phi(x_i) - \mu_k)(\phi(x_i) - \mu_k)^\top \right] \to 0.$$

(NC2) Simplex Mean Structure: The class means, after centering by the global mean $\mu_G = \frac{1}{K} \sum_k \mu_k$, converge to the vertices of a simplex ETF:

$$\mu_k - \mu_G \in \text{ETF structure}, \quad \forall k \in \{1, \ldots, K\}.$$

(NC3) Feature-Weight Alignment: The classifier weights w_k also form a simplex ETF and align directionally with their respective class means:

$$\mu_k - \mu_G \parallel w_k - \bar{w}, \quad \text{where } \bar{w} = \frac{1}{K} \sum_k w_k.$$

(NC4) Nearest-Class Decision Rule: When (NC1)–(NC3) are satisfied, the network's prediction reduces to assigning the sample to the nearest class mean (or equivalently, nearest weight vector) in the inner product sense:

$$\hat{y} = \arg\max_k \langle w_k, \phi(x) \rangle = \arg\min_k \|\phi(x) - \mu_k\|^2.$$

Overall, Neural Collapse represents a regime in which both the learned features and classifier weights exhibit optimal geometric alignment, achieving maximal inter-class separability while minimizing intra-class variability. This structure is theoretically linked to the maximization of the Fisher Discriminant Ratio, providing a compelling explanation for the generalization performance of overparameterized models.

3.1.3 Baseline: NC-FSCIL

In our study, we adopt the NC-FSCIL method as a representative geometry-aware baseline for the few-shot class-incremental learning (FSCIL) task. NC-FSCIL is based on the neural collapse phenomenon and employs a fixed classifier structure derived from a pre-assigned simplex Equiangular Tight Frame (ETF) spanning all classes across sessions.

Let the base session contain K_0 classes, and each of the T incremental sessions introduce p new classes, resulting in a total of $K = K_0 + T \cdot p$ classes. The classifier is constructed as a matrix $\widehat{\mathbf{W}}_{\text{ETF}} \in \mathbb{R}^{d \times K}$, whose column vectors correspond to class prototypes. Following the ETF construction, each pair of prototypes $(\widehat{\mathbf{w}}_{k_1}, \widehat{\mathbf{w}}_{k_2})$ satisfies:

$$\langle \widehat{\mathbf{w}}_{k_1}, \widehat{\mathbf{w}}_{k_2} \rangle = \begin{cases} 1, & \text{if } k_1 = k_2, \\ -\frac{1}{K-1}, & \text{if } k_1 \neq k_2. \end{cases} \tag{3.3}$$

The model consists of a backbone network f and a projection layer g. Given an input x_i, the model extracts an intermediate feature $h_i = f(x_i)$, projects it via g, and normalizes the result to obtain:

$$\tilde{\boldsymbol{\mu}}_i = g(h_i), \quad \widehat{\boldsymbol{\mu}}_i = \frac{\tilde{\boldsymbol{\mu}}_i}{\|\tilde{\boldsymbol{\mu}}_i\|_2}. \tag{3.4}$$

Training is performed using the *dot-regression loss*, which directly aligns the normalized feature with its corresponding ETF prototype. Specifically, for a sample x_i with ground-truth label y_i, the loss is defined as:

$$\mathcal{L}\left(\widehat{\boldsymbol{\mu}}_i, \widehat{\mathbf{W}}_{\mathrm{ETF}}\right) = \frac{1}{2}\left(\widehat{\mathbf{w}}_{y_i}^{\top}\widehat{\boldsymbol{\mu}}_i - 1\right)^2, \tag{3.5}$$

where $\widehat{\mathbf{w}}_{y_i}$ denotes the fixed ETF prototype for class y_i.

During the base session ($t = 0$), both f and g are optimized to minimize the average DR loss. In incremental sessions ($t > 0$), f is frozen and only g is fine-tuned using the new class data along with a memory buffer that stores the mean intermediate features of old classes. Classification at test time is performed via inner product between the normalized output feature and the fixed ETF prototypes.

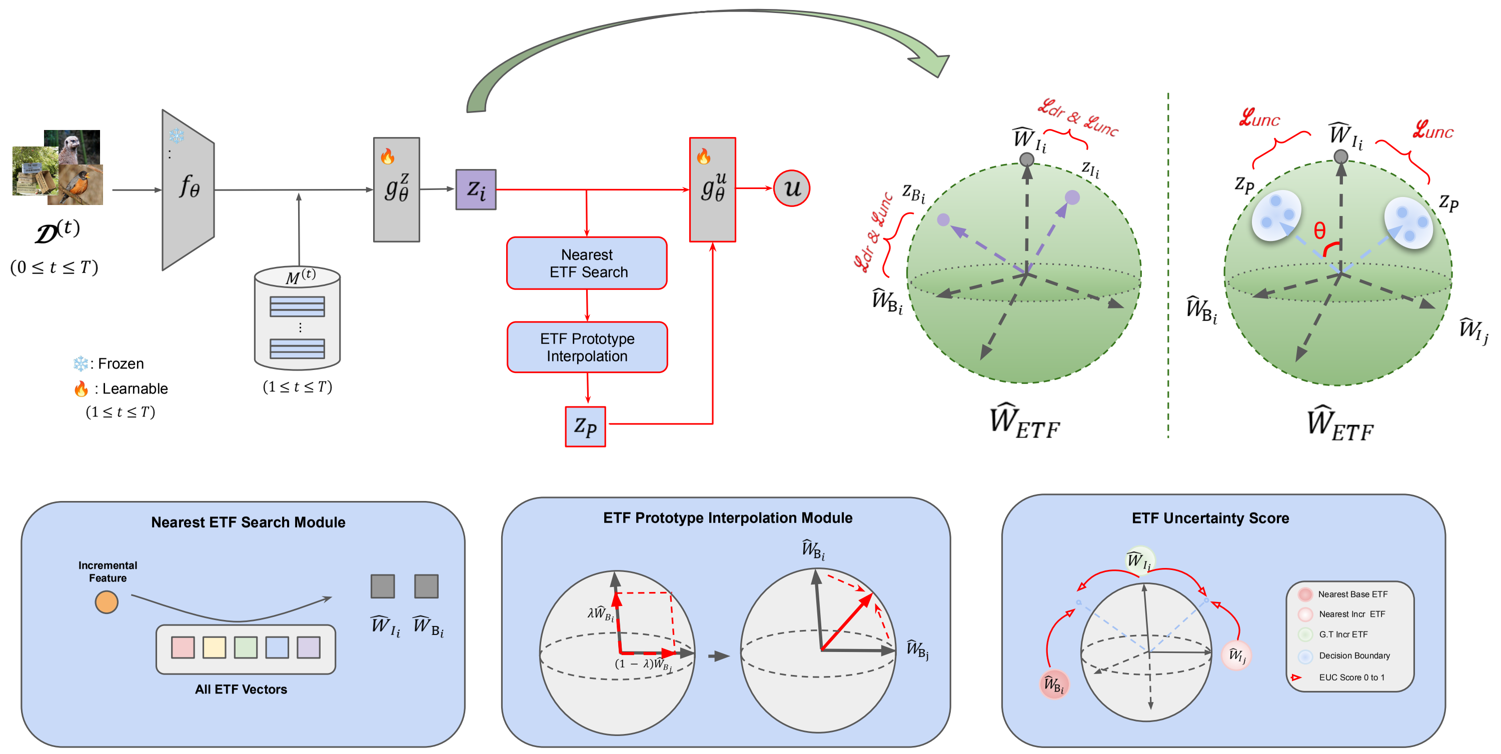


FIGURE 3.1: Overall architecture of proposed Neural Collapse-Driven, Uncertainty-aware Few-shot Class-Incremental Learning. Red line depicts our module on top of the baseline.

3.2 Our Method

3.2.1 Neural Collapse-Driven Uncertainty Estimation

Based on the NC-FSCIL baseline, we propose a neural collapse-driven framework for estimating the uncertainty of feature representations in few-shot class-incremental learning. Our goal is to predict a scalar uncertainty score $u \in [0, 1]$ for a given normalized representation $z \in \mathbb{R}^d$ in a supervised manner by leveraging the geometric structure of pre-defined ETF vectors.

As illustrated in Figure 3.1, our model consists of three components: a backbone encoder f, a projection head for representation g^z, and another projection head for uncertainty g^u. Given an input x_i, the backbone network f extracts intermediate features, which are then projected via g^z and normalized to yield the representation $\widehat{\mu}_i$. This representation is further passed into g^u, which predicts the uncertainty score $\hat{u}_i$.

To supervise uncertainty learning, we define an *EUC Score* function that reflects the relative alignment of z with the base and incremental ETF vectors:

$$\text{EUC-Score} = \frac{1 + [\max\cos(f, W_{\text{base}}) - \max\cos(f, W_{\text{incr}})]}{2}$$

This score lies in the range $[0, 1]$, where a lower score indicates strong alignment with the incremental ETF prototypes, and a higher score indicates strong alignment with the base ETF prototypes. Representations near the decision boundary, equidistant from both prototypes, are assigned scores close to 0.5. This formulation produces a smooth transition of uncertainty from incremental-aligned features (scores near 0) to base-aligned features (score near 1), providing an interpretable measure of directional confidence.

We supervise the uncertainty projection module g^u using EUC scores for both base and incremental session data. To supervise the uncertainty projection module g^u, we define the following regression loss between the predicted uncertainty score $\hat{u}_i = g^u(\widehat{\mu}_i)$ and the EUC score:

$$\mathcal{L}_{\text{unc}} = \text{MSE}(g^u(\widehat{\mu}_i),\ \text{EUC-Score}(\widehat{\mu}_i)), \tag{3.6}$$

where $\widehat{\mu}_i$ is the normalized feature representation for input x_i.
However, due to the strong alignment induced by the DR loss, all real features tend to be tightly clustered near their respective ETF prototypes. This results in a behavior in g^u, where it collapses to predicting near-zero or near-one uncertainty scores for all inputs, where the model fails to produce meaningful variation across inputs. In order to provide uncertainty supervision with greater variance and better coverage across the [0,1] range, we generate *pseudo samples* between ETF prototypes to provide richer supervision and guide g^u to learn more discriminative uncertainty estimates. We use two modules to create the samples:

Nearest ETF Search Module. Given a feature z_k, we retrieve the ETF vector with the highest cosine similarity:

$$\mathbf{w}^{\text{nearest}} = \arg \max_{\mathbf{w} \in \widehat{\mathbf{W}}_{\text{ETF}}} \cos(z_k, \mathbf{w}). \tag{3.7}$$

In the base session ($t = 0$), we retrieve the ground-truth ETF vector $\mathbf{w}_k$ for the given label, and the most similar incremental ETF vector $\mathbf{w}_i$ based on cosine similarity:

$$\mathbf{w}_i = \arg \max_{\mathbf{w} \in \widehat{\mathbf{W}}_{\text{Incr}}} \cos(z_k, \mathbf{w}), \quad \mathbf{w}_k = \widehat{\mathbf{W}}_{\text{ETF}}[y_k],$$

where i denotes the index of the nearest incremental ETF vector, and k is the index corresponding to the ground-truth label y_k.

In incremental sessions ($t > 0$), we retrieve both the most similar base and incremental ETF vectors:

$$\mathbf{w}_i = \arg \max_{\mathbf{w} \in \widehat{\mathbf{W}}_{\text{Base}}} \cos(z_k, \mathbf{w}), \quad \mathbf{w}_j = \arg \max_{\mathbf{w} \in \widehat{\mathbf{W}}_{\text{Incr}}} \cos(z_k, \mathbf{w}), \quad \mathbf{w}_k = \widehat{\mathbf{W}}_{\text{ETF}}[y_k],$$

where i and j are the indices of the most similar base and incremental ETF vectors, respectively, and k is the index for the ground-truth label.

These selected ETF vectors form the interpolation triplet:

$$\{\mathbf{w}_k,\ \mathbf{w}_i,\ \mathbf{w}_j\}.$$

ETF Prototype Interpolation Module. Using the retrieved ETF vectors, we generate pseudo samples via interpolation. One element of the interpolation pair is fixed as the ground-truth ETF vector $\mathbf{w}_k$. In the base session ($t = 0$), the other vector is set to the nearest incremental ETF vector $\mathbf{w}_j$. In incremental sessions ($t > 0$), we use both the nearest base ETF vector $\mathbf{w}_i$ and the nearest incremental ETF vector $\mathbf{w}_j$

Algorithm 1 Pseudo-code for ETF-based Uncertainty Estimation (PyTorch style)

```
# etf_vec: [D, C] ETF classifier matrix
# z: feature representation [B, D]
# gt_label: ground-truth labels [B]
# g^u: uncertainty MLP head
# pseudo_ratio: float in (0, 1)

# --- Step 1: Compute Uncertainty Score for Real Samples ---
z = pre_logits(z)  # project features
w_base = etf_vec[:, :C_base]       # base ETF
w_incr = etf_vec[:, C_base:]       # incremental ETF

sim_base = torch.matmul(z, w_base)    # [B, C_base]
sim_incr = torch.matmul(z, w_incr)    # [B, C_incr]

max_base = sim_base.max(dim=-1).values    # [B]
max_incr = sim_incr.max(dim=-1).values    # [B]

euc_score = 0.5 * (1 + (max_base - max_incr))  # [B]
euc_score = torch.clamp(euc_score, 0, 1)

# --- Step 2: Real Sample Uncertainty Loss ---
score_pred = g_u(z)
loss_real = MSE(score_pred, euc_score)

# --- Step 3: Generate Pseudo Samples between ETF Prototypes ---
pseudo_feats = []
for i in range(B):
    label = gt_label[i]
    if label < C_base:
        w_gt = w_base[:, label]
        w_other = find_nearest(w_gt, w_incr)  # cosine similarity
        pairs = [w_other]
    else:
        w_gt = w_incr[:, label - C_base]
        w_b = find_nearest(w_gt, w_base)
        w_i = find_nearest(w_gt, w_incr, exclude=label - C_base)
        pairs = [w_b, w_i]

    for w_other in pairs:
        for alpha in interpolation_ratios:
            pseudo = (1 - alpha) * w_gt + alpha * w_other
            noise = torch.randn_like(pseudo) * sigma
            pseudo = F.normalize(pseudo + noise, dim=0)
            pseudo_feats.append(pseudo)

pseudo_feats = torch.stack(pseudo_feats)  # [B', D]

# --- Step 4: Pseudo Sample Uncertainty Supervision ---
sim_base_pseudo = torch.matmul(pseudo_feats, w_base)  # [B', C_base]
sim_incr_pseudo = torch.matmul(pseudo_feats, w_incr)  # [B', C_incr]

max_base_pseudo = sim_base_pseudo.max(dim=-1).values
max_incr_pseudo = sim_incr_pseudo.max(dim=-1).values

score_target_pseudo = 0.5 * (1 + (max_base_pseudo - max_incr_pseudo))
score_target_pseudo = torch.clamp(score_target_pseudo, 0, 1)

score_pred_pseudo = g_u(pseudo_feats)
loss_pseudo = MSE(score_pred_pseudo, score_target_pseudo)
```

to generate two separate pseudo samples for each feature. Each pseudo representation is computed as:

$$\tilde{z}_{\text{pseudo}} = \alpha \cdot \mathbf{w}_k + (1-\alpha) \cdot \mathbf{w}_{\text{near}}, \quad \alpha \in (0, 0.5), \tag{3.8}$$

where $\mathbf{w}_{\text{near}} \in \{\mathbf{w}_j\}$ in the base session, and $\mathbf{w}_{\text{near}} \in \{\mathbf{w}_i, \mathbf{w}_j\}$ in incremental sessions. The target uncertainty score for each pseudo sample $\tilde{z}_{\text{pseudo}}$ is computed using the EUC-Score function defined in Eq. 4.5, in the same way as for real samples.

To promote diversity and improve generalization, we additionally inject Gaussian noise into the pseudo samples:

$$\hat{z}_{\text{pseudo}} = \tilde{z}_{\text{pseudo}} + \delta, \quad \delta \sim \mathcal{N}(0, \sigma^2 \mathbf{I}). \tag{3.9}$$

These pseudo samples and their corresponding EUC scores are used to supervise g^u alongside real samples. This enables the uncertainty head to learn meaningful gradients even in regions that are sparsely populated in the training data, particularly near the decision boundary between base and incremental classes.

3.2.2 Uncertainty Score based Calibration

In Few-Shot Class-Incremental Learning (FSCIL), base classes are trained with abundant labeled data, whereas incremental classes are introduced with only a few labeled samples per class. This severe data imbalance causes the model to exhibit higher prediction uncertainty and a higher misclassification rate on incremental classes. To alleviate this issue, we propose an uncertainty-aware calibration method that adjusts the cosine similarity-based classification scores using an uncertainty-based scaling factor.

Cosine Similarity-Based Prediction. In FSCIL, classification is typically performed by computing the cosine similarity between the normalized feature representation f and each class weight vector w_j in a fixed Equiangular Tight Frame (ETF) classifier. The predicted label is obtained by:

$$\hat{y} = \arg\max_j \cos(f, w_j)$$

However, such predictions do not account for the model's uncertainty, especially for incremental classes where the confidence is inherently lower.

Uncertainty Score Computation. We define an uncertainty score $u(x)$ for a given sample x as the normalized difference between its maximum cosine similarity with the incremental class weights W_{incr} and the base class weights W_{base}:

$$u(x) = \frac{1 + [\max\cos(f, W_{\text{base}}) - \max\cos(f, W_{\text{incr}})]}{2}$$

This score is high when the sample lies near the decision boundary or when its alignment with both base and incremental classes is ambiguous.

Likelihood-Based Scaling. During each incremental session $t > 0$, we estimate the Gaussian distribution of uncertainty scores for both base and incremental groups using training data. For base classes, the prototype memory is used to compute class-wise uncertainty statistics, while incremental classes use the available few-shot samples. Given a test sample x, its uncertainty likelihood $\lambda(x)$ is computed under the corresponding distribution as:

$$\lambda_{\text{base}} = \mathcal{N}(1 - u(x); \mu_b, \sigma_b^2), \quad \lambda_{\text{incr}} = \mathcal{N}(u(x); \mu_i, \sigma_i^2)$$

where μ and σ^2 are the mean and variance of the estimated uncertainty distribution for the predicted group (base or incremental). To ensure a consistent interpretation of uncertainty across all classes, we apply a transformation of $1 - u(x)$ for base samples, since the uncertainty scoring function is designed to assign values close to 1 for base-aligned features. This transformation guarantees higher values consistently reflect high uncertainty across both base and incremental classes.

Score Calibration. Finally, we apply the normalized confidence weight $w(x)$ to adjust the cosine similarity scores. The calibrated score $\tilde{s}_j$ for class j is defined as:

$$\tilde{s}_j = \cos(f, w_j) + \alpha * w(x),$$

where $w(x) = \frac{\lambda_{\text{base}}}{\lambda_{\text{base}} + \lambda_{\text{incr}}}$ is the normalized likelihood-based confidence weight derived from the uncertainty scores.

Algorithm 2 Post-hoc Uncertainty-Aware Score Calibration

Input: Test feature $f(x)$, ETF weights W_b, W_i, group-wise uncertainty statistics $(\mu_b, \sigma_b^2), (\mu_i, \sigma_i^2)$
Output: Calibrated class scores $\tilde{s}_j$

[1. Cosine Similarity Computation]

Compute cosine similarities between $f(x)$ and base/incremental class weights:

$$\max \cos(f, W_b), \quad \max \cos(f, W_i)$$

[2. Uncertainty Score Estimation]

Define post-hoc uncertainty score as:

$$u(x) = \frac{1 + [\max \cos(f, W_b) - \max \cos(f, W_i)]}{2}$$

[3. Likelihood Computation]

Compute likelihood of $u(x)$ under both base and incremental uncertainty distributions:

$$\lambda_{\text{base}} = \mathcal{N}(1 - u(x); \mu_b, \sigma_b^2), \quad \lambda_{\text{incr}} = \mathcal{N}(u(x); \mu_i, \sigma_i^2)$$

[4. Normalized Confidence Weight]

Define confidence weight $w(x)$ for calibration as:

$$w(x) = \frac{\lambda_{\text{base}}}{\lambda_{\text{base}} + \lambda_{\text{incr}}}$$

[5. Score Calibration]

Apply $w(x)$ to adjust cosine scores:

$$\tilde{s}_j = \cos(f, w_j) + \alpha * w(x)$$

Predict class label:

$$\hat{y} = \arg\max_j \tilde{s}_j$$

Chapter 4

Experiments

In this section, we first describe the datasets used for FSCIL in Section 4.1, followed by evaluation metric used in FSCIL in Secition 4.2. Then, we provide detailed training configurations and implementation details in Section 4.3. Experimental results are presented in Section 4.4, followed by an in-depth discussion and analysis in Section 4.5.

4.1 Dataset

CIFAR100 [38] is a widely-used dataset comprising 60,000 color images (32$\times$32 resolution) spanning 100 distinct object categories. Each class is composed of 500 training and 100 testing examples. Following prior FSCIL benchmarks [14], [16], [25], we divide the dataset into 60 base classes and 40 novel classes. The novel classes are further partitioned into 8 incremental sessions, with each session containing 5 new classes and only 5 labeled samples per class (i.e., 5-way 5-shot learning).

***mini*ImageNet** is a subset of ImageNet [39] containing 100 classes, each with 600 images. All images are resized to 84$\times$84. We adopt the conventional FSCIL split: 60

classes for base training and the remaining 40 classes for incremental learning. The 40 incremental classes are evenly distributed across 8 sessions, where each session includes 5 novel classes with 5 training samples per class.

Caltech-UCSD Birds-200-2011 (CUB200) [40] is a fine-grained classification dataset with 11,788 images across 200 bird species. Following the setting in previous literature, we use 100 classes for base training and the remaining 100 classes for incremental sessions. These 100 incremental classes are split into 10 sessions, each comprising a 10-way 5-shot classification task. All images are resized to 224×224 resolution for training and evaluation.

Architectures Prior studies widely adopt ResNet-12, ResNet-18, and ResNet-20 [1] for FSCIL experiments. We use ResNet-12 following [25]. For CUB-200, we use ResNet-18 (pre-trained on ImageNet) following other studies. We adopt a two-layer MLP block as the projection layer following [25] and add additional two-layer MLP block for our uncertainty module.

4.2 Evaluation Metric

To enable fair comparison with previous FSCIL methods, we use *average accuracy* (*aAcc*) and *generalized average accuracy* (*gAcc*) as evaluation metrics.

The equation for *aAcc* is as follows:

$$aAcc_i = \frac{\frac{|\mathcal{Y}_i|}{|\mathcal{Y}_{\text{novel}}|} A_i^1 + \sum_{j=2}^{i} A_i^j}{\frac{|\mathcal{Y}_i|}{|\mathcal{Y}_{\text{novel}}|} + (i-1)} \tag{4.1}$$

$|\mathcal{Y}_i|$ is the size of the labeling space of task $\mathcal{T}_i$, A_i^j denotes the accuracy on the class set of task $\mathcal{T}_j$, when the model has been trained up to the task $\mathcal{T}_i$. In FSCIL settings, the class size ratio $\frac{|\mathcal{Y}_i|}{|\mathcal{Y}_{\text{novel}}|}$ is often large (e.g., 12 for CIFAR-100 and miniImageNet, 10 for CUB-200). As a result, existing metrics such as *aAcc* and *last-task average accuracy* (*lAcc*) tend to be dominated by the base class performance. Even when the model performs poorly on the novel classes, these metrics still show a high value due to the high weight of base accuracy. To overcome this limitation, Yourself [24] defined a generalized accuracy metric *gAcc* as follows:

$$gAcc_i(\alpha) = \frac{\alpha \cdot \frac{|\mathcal{Y}_i|}{|\mathcal{Y}_{\text{novel}}|} A_i^1 + \sum_{j=2}^{i} A_i^j}{\alpha \cdot \frac{|\mathcal{Y}_i|}{|\mathcal{Y}_{\text{novel}}|} + (i-1)} \tag{4.2}$$

gAcc generalizes the weighting factor α to any value in $[0, 1]$. The area under the curve (AUC) of $gAcc_i(\alpha)$ is computed as:

$$gAcc_i = \int_0^1 gAcc_i(\alpha)\, d\alpha \tag{4.3}$$

For multiple sessions, the overall generalized accuracy is defined as the average over all sessions:

$$gAcc = \frac{1}{n_t} \sum_i gAcc_i \tag{4.4}$$

4.3 Training implementation

We adopt distributed data parallelism for training, computing the overall loss using the aggregated global batch across all GPUs. Batch normalization layers are synchronized to maintain consistent statistics across devices. We closely follow the training configuration and hyperparameters used in NC-FSCIL [25]. In the base session, we use a batch size of 256. For each incremental session, we adopt a batch size of 32, which includes both new class samples and previously stored intermediate features from memory. This differs from the original baseline implementation, which used a smaller batch size of 8 and leveraged data-parallel training over 8 GPUs. In contrast, our implementation uses 2 GPUs.

For ***mini*ImageNet**, the base session is trained for 500 epochs, while each incremental session is trained for 100 to 170 iterations. The learning rate is initialized to 0.25 for the base session and 0.025 for incremental updates. On **CIFAR-100**, we train for 200 epochs in the base session, and each incremental session is trained for 50 to 200 iterations. A fixed initial learning rate of 0.25 is used for both phases. For **CUB-200**, the base session consists of 80 training epochs, and each incremental session runs for 105 to 150 iterations. Learning rates are set to 0.025 for the base session and 0.05 for the incremental sessions. In all experiments, we employ cosine annealing for learning rate scheduling and use SGD with momentum as the optimizer. Our training platform consists of two RTX 3090 GPUs and an 8-core Intel(R) Core(TM) i9-11900K CPU running at 3.50GHz. To ensure reproducibility, all experiments are conducted with a fixed random seed. For the additional MLP used in our framework, we adopt a two-layer architecture with a hidden dimension of 256. A ReLU activation function is applied after the first layer, followed by a sigmoid activation in the final layer to produce output values in the range of 0 to 1. In our pseudo-sample generation module, we linearly interpolate between two ETF prototypes using interpolation ratios ranging from 0.0 to 1.0 in steps of 0.05. To enhance diversity, Gaussian noise with a standard deviation of 0.05 is added to the interpolated features. We set the weight of the uncertainty loss to 0.1 throughout all training stages.

Method	Accuracy in each session (%) ↑									*aAcc*	*gAcc*
	0	1	2	3	4	5	6	7	8		
NC-FSCIL (Baseline)	82.33	77.09	72.86	68.59	64.94	61.15	59.38	56.58	54.86	66.42	56.99
Base Class Acc.	82.33	79.72	78.70	77.28	76.55	74.08	73.20	72.13	71.85	-	-
Incr Class Acc.	-	45.60	37.80	33.80	30.10	30.12	31.73	29.91	29.37	-	-
same hyperparameter	82.33	76.12	71.59	67.41	64.35	60.53	57.71	55.77	53.67	65.49	54.92
Base Class Acc.	82.33	81.88	80.60	78.93	78.08	76.35	77.03	77.23	76.03	-	-
Incr Class Acc.	-	7.00	17.50	21.33	23.15	22.56	19.07	17.98	20.13	-	-
Ours											
+ UC module	82.88	77.62	73.27	67.32	64.72	62.11	59.31	57.21	54.71	66.57	57.69
+ UC Calibration	82.88	**77.25**	**72.90**	66.41	64.74	**61.91**	59.21	**57.02**	**54.47**	66.24	**58.22**
Base Class Acc.	82.88	81.02	79.37	74.77	78.53	74.72	74.42	74.92	72.73	-	-
Incr Class Acc.	–	34.20	35.60	33.00	23.35	31.16	28.80	26.34	27.07	-	-

TABLE 4.1: Comparison with the baseline on CIFAR100 dataset. We also report base vs. incremental class average accuracy at each method. The results of NC-FSCIL are obtained by our own reproduction.

4.4 Results

4.4.1 Cifar100 Result

Table 4.1 presents the performance comparison on CIFAR100 between our proposed method and the NC-FSCIL baseline. Since the original results of NC-FSCIL could not be fully reproduced due to differences in GPU count and associated hyperparameters, we report both the best reproduced results and the ones obtained under the same hyperparameter settings for a consistent comparison. Our first observation is that incorporating an additional MLP for uncertainty prediction does not significantly degrade the baseline performance. In fact, the average *aAcc* and *gAcc* slightly increase by 0.15% and 0.70%, respectively, compared to the baseline. This suggests that the learned representations retain their discriminative power while modeling uncertainty effectively. After applying our calibration strategy, we observe a notable improvement in *gAcc*, achieving 1.23% higher performance than

Method	Accuracy in each session (%) ↑											*aAcc*	*gAcc*
	0	1	2	3	4	5	6	7	8	9	10		
NC-FSCIL (Baseline)	80.45	75.98	72.30	70.28	68.17	65.16	64.43	63.25	60.66	60.01	59.44	67.28	61.56
Base Class Acc.	80.45	76.89	77.62	78.63	77.23	77.13	76.85	76.29	76.61	75.94	76.19	-	-
Incr Class Acc.	-	66.67	45.41	42.59	45.88	41.72	44.11	44.99	41.16	42.66	43.07	-	-
Ours													
+ UC module	80.87	76.01	72.97	69.93	67.83	65.44	64.87	63.71	61.47	60.36	59.67	67.55	61.86
+ UC Calibration	80.87	75.63	**72.97**	70.14	67.95	**65.44**	**65.04**	**63.73**	**61.22**	60.36	**60.06**	**67.30**	**62.55**
Base Class Acc.	80.87	75.91	77.44	77.86	76.19	76.50	75.84	75.56	75.28	74.55	74.97	-	-
Incr Class Acc.	-	72.76	50.35	44.56	47.68	43.78	47.37	47.19	44.02	44.90	45.49	-	-

TABLE 4.2: Comparison with the baseline on the CUB dataset. We also report base vs. incremental class average accuracy at each method.

the best baseline result. Moreover, except for sessions 3, 4, and 6, our method consistently outperforms the baseline in all sessions. When comparing results under the same hyperparameter configuration, our method exhibits slightly lower base class accuracy, but significantly higher incremental class accuracy. Specifically, in the final session (session 8), the base class accuracy drops by 3.3%, while the incremental class accuracy improves by 6.96%, ultimately leading to better overall performance in both *aAcc* and *gAcc*.

4.4.2 CUB200 Result

Table 4.2 presents a session-wise comparison between our method and the NC-FSCIL baseline on the CUB-200 dataset. For this benchmark, we report the baseline performance as presented in the original NC-FSCIL paper. First, we observe that adding the uncertainty (UC) module yields a slight improvement in average accuracy, suggesting that our model is capable of producing representations that also encode uncertainty information without degrading discriminative power. Second, we observe consistent improvements in incremental class performance across all sessions after applying our uncertainty-based calibration. Specifically, the average base class accuracy across all sessions decreases only by 0.71%, while the average

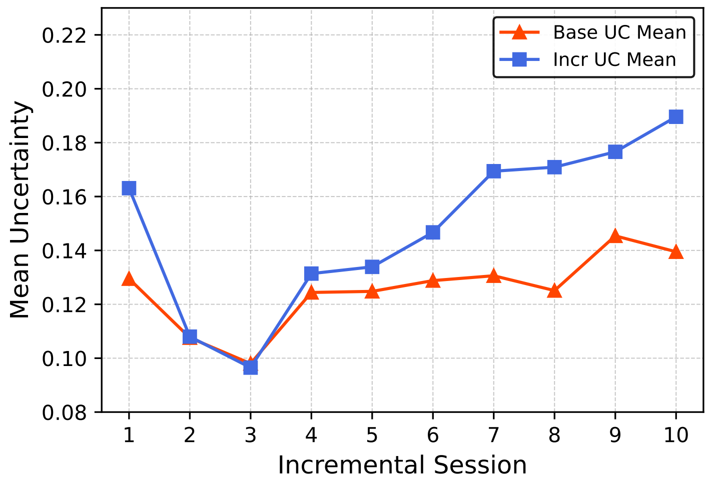


FIGURE 4.1: Mean uncertainty of the training set for each incremental session (1 to 10) on the CUB-200 dataset. The mean uncertainty of base class samples remains consistently lower than that of incremental class samples across all sessions.

incremental class accuracy improves by 2.46%. As a result, the overall accuracy (*aAcc*) increases by 0.17%, and the generalized accuracy (*gAcc*) shows a notable improvement of 1.03%. Overall, although this calibration incurs a slight decrease in base class accuracy, it significantly enhances incremental class accuracy by effectively reducing their misclassification into base classes. As illustrated in Fig. 4.1, the average uncertainty scores of base class samples remain consistently lower than those of incremental class samples throughout all sessions. This discrepancy enables our calibration strategy to assign larger correction weights to incremental samples, thereby refining their decision scores.

Method	Accuracy in each session (%) ↑									*aAcc*	*gAcc*
	0	1	2	3	4	5	6	7	8		
NC-FSCIL (Baseline)	84.02	76.80	72.00	67.83	66.35	64.04	61.46	59.54	58.31	67.81	60.93
Base Class Acc.	82.33	78.68	75.98	74.10	74.95	76.18	76.18	75.55	76.30	-	-
Incr Class Acc.	-	54.20	48.10	42.73	40.55	34.88	32.00	32.09	31.33	-	-
Ours											
+ UC module	84.33	76.35	72.71	68.23	67.11	64.41	62.67	60.39	59.01	<u>68.35</u>	61.77
+ UC Calibration	84.33	75.28	**72.23**	67.76	**66.47**	63.67	**62.54**	**60.45**	**59.18**	**67.99**	**61.96**
Base Class Acc.	84.33	77.07	76.03	73.67	74.47	74.22	75.10	75.15	75.47	-	-
Incr Class Acc.	–	53.80	49.40	44.13	42.50	38.36	37.43	35.26	34.75	-	-

TABLE 4.3: Comparison with the baseline on *mini*ImageNet dataset. We also report base vs. incremental class average accuracy at each method.

4.4.3 *Mini*ImageNet Result

Table 4.3 reports a session-wise performance comparison between our method and the NC-FSCIL baseline on the *mini*ImageNet dataset. As with CUB-200, the baseline results are taken directly from the original NC-FSCIL publication. Initially, we find that incorporating the uncertainty (UC) module slightly improves the average accuracy. Moreover, the application of our uncertainty-driven calibration method leads to consistent performance gains for incremental classes across all sessions. While there is a minor drop in base class accuracy, the calibration process substantially reduces confusion between incremental and base classes, resulting in a meaningful improvement in incremental class classification. In the final session, the base class accuracy decreased by only 0.83%, whereas the incremental class accuracy increased by 3.42%. Consequently, our method outperforms the baseline in terms of *aAcc* by 0.18% and *gAcc* of 1.03%. Overall, these results validate that our uncertainty-aware calibration framework generalizes well across datasets, offering robust improvements in incremental learning scenarios with minimal cost to base class performance.

TABLE 4.4: Comparison of Estimated(Est.) UC Score and Post-Hoc Scores across datasets.

(A) CIFAR100

	Est.	Post-Hoc
Base	$\mathcal{N}(0.2857, 0.084)$	$\mathcal{N}(0.23, 0.054)$
Incr	$\mathcal{N}(0.3474, 0.083)$	$\mathcal{N}(0.44, 0.05)$

(B) CUB200

	Est.	Post-Hoc
Base	$\mathcal{N}(0.33, 0.03)$	$\mathcal{N}(0.312, 0.013)$
Incr	$\mathcal{N}(0.269, 0.026)$	$\mathcal{N}(0.426, 0.008)$

(C) MiniImageNet

	Est.	Post-Hoc
Base	$\mathcal{N}(0.227, 0.04)$	$\mathcal{N}(0.306, 0.022)$
Incr	$\mathcal{N}(0.403, 0.04)$	$\mathcal{N}(0.449, 0.019)$

4.5 Discussion

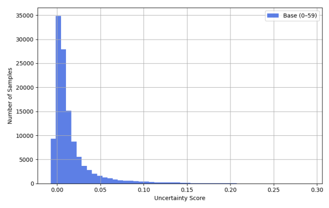
4.5.1 Accuracy and Effectiveness of Uncertainty Module

In this section, we analyze how accurately the uncertainty module has been trained—specifically, how closely the model-predicted uncertainty aligns with the ground-truth (post-hoc) uncertainty scores. Table 4.4 presents the distributional differences between the estimated and post-hoc uncertainty scores, evaluated on the test sets of CIFAR100, CUB200, and *mini*ImageNet. The results are reported separately for base and incremental classes. Although the predicted uncertainty does not perfectly match the post-hoc ground-truth, we observe that on CIFAR100 and *mini*ImageNet, the model successfully captures the intended tendency: base class samples consistently exhibit lower uncertainty than incremental class samples. In contrast, performance on CUB200 is relatively weaker. We conjecture that this is due to the fine-grained nature of the CUB200 dataset and its significantly smaller training size (only 5,000 images), compared to CIFAR100 (30,000 images) and *mini*ImageNet (36,000 images), which makes it more challenging for the model to learn reliable uncertainty estimates. The current uncertainty module is implemented as a 2-layer MLP with a hidden dimension of 256 and a sigmoid activation at the final layer. While this simple architecture yields meaningful trends, further ablation studies are required to investigate whether a more sophisticated design can provide more precise and robust uncertainty estimates.

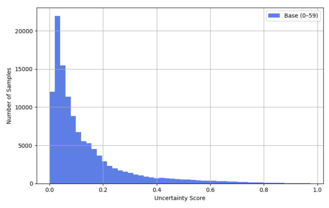
We also discuss the effectiveness of our uncertainty module. As shown in all three benchmark tables (Table 4.1, Table 4.2, and Table 4.3), we observe a slight improvement in the overall accuracy when the uncertainty (UC) module is applied alone. We conjecture that this is because the addition of the UC loss to the existing DR loss encourages the model to learn a richer representation. While the DR loss primarily focuses on aligning feature representations with specific ETF vectors, the UC loss aims to model the relative alignment between base and incremental class ETF groups. These two objectives are subtly different. For instance, consider two feature vectors A and B that form the same angle with a particular ETF vector. Their DR losses would be identical, but their EUC scores could differ. This suggests that the uncertainty module captures additional discriminative information not reflected in the DR loss alone. Further mathematical analysis of this difference may provide deeper insights and theoretical justification for the effectiveness of the uncertainty module, which we leave for future work.

4.5.2 EUC Score Distribution

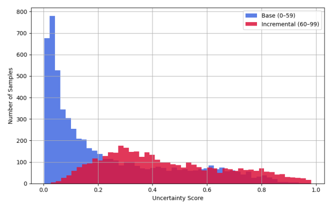
In the base session, the model is trained on a large amount of base class data, and the Dot Regression loss (DR loss) enforces strong alignment with the ground-truth base ETF vectors. As a result, immediately after the base session, the uncertainty scores of the base class training samples are heavily concentrated near zero. As



(A) Session 0



(B) Session 9



(C) Session 9

FIGURE 4.2: Comparison of base and incremental uncertainty distributions on CIFAR100. (A) and (B) show the train set distributions, while (C) shows the test set distribution.

shown in Figure 4.2a, the distribution has a mean of 0.133 and a variance of 0.022, indicating a sharp peak close to zero. In contrast, Figure 4.2b illustrates how this distribution shifts as incremental sessions progress. Due to catastrophic forgetting, the features of base class samples gradually deviate from their original ETF vectors. In the final session, as shown in Figure 4.2b, the distribution becomes noticeably skewed to the right, with a mean of 0.230 and a variance of 0.054. Lastly, Figure 4.2c presents the uncertainty score distributions on the CIFAR100 test set, comparing 6,000 base samples from the base session and 4,000 incremental samples from the incremental session. As expected, incremental class samples exhibit higher uncertainty scores than base classes, since they are influenced by both base and incremental ETF vectors.

4.5.3 Discrepancy Between Train and Test Uncertainty Distributions

In FSCIL settings, it is evident that the true distribution of uncertainty scores over the test set differs from that of the training data used for calibration. However, due to the inherent constraints of FSCIL—where previously seen training samples cannot be used in incremental sessions—we are unable to directly access the true distribution of earlier class features. To address this, we leverage the idea of a common practice in prior FSCIL studies [24], [25], which use memory-based replay to mitigate catastrophic forgetting. In our case, we adopt a memory buffer that stores prototype vectors for each previously learned class. During each incremental session, we estimate the uncertainty distributions for base classes using these stored prototypes. Specifically, we calculate the mean and variance of their uncertainty scores based on the model's predictions over the prototypes. For the current session's new classes, we compute the mean and variance using the few-shot training samples available (e.g., 5 classes × 5 shots = 25 images in CIFAR100). These statistics are then used to fit Gaussian distributions that approximate the uncertainty score distributions for both base and incremental class groups. Although we cannot access the actual test-time uncertainty distribution, we can treat the test set as an ideal reference, serving as an upper bound. While our method does not reach this upper bound, it demonstrates that a practical approximation is achievable under the constraints of FSCIL. Future work may explore strategies to better approximate or even approach this upper bound through more accurate estimation techniques.

4.5.4 Other Uncertainty Score Function

In addition to the subtraction-based formulation for uncertainty scoring, we also experimented with an alternative ratio-based definition, as described in Equation 4.5. Unlike the original formulation, which quantifies the relative influence of the base and incremental prototypes via their difference, this variant expresses their contributions as a normalized ratio. In this formulation, features that are strongly aligned with either the base or incremental ETF vector yield low uncertainty scores, while features near the decision boundary-i.e., those equidistant from both directions- produce scores approaching 1. A key distinction from the subtraction-based method lies in its directional behavior: in the original formulation, uncertainty tends to increase as features shift from alignment with the incremental ETF to the base ETF. By contrast, the ratio-based score increases when features are equally distant from both prototypes, emphasizing ambiguity at the boundary. However, we observed that this ratio-based formulation often results an instability during training. Specifically, the uncertainty head frequently collapsed to output a near-constant value during incremental sessions, failing to capture meaningful uncertainty variations. Consequently, we do not adopt this variant in our final implementation.

$$\text{EUC-Score} = \min\left(\max\left(0, \min\left(\frac{\max\cos_i(z_k, W^i_{\text{Incr}}) + \frac{1}{C-1}}{\max\cos_j(z_k, W^j_{\text{Base}}) + \frac{1}{C-1} + \varepsilon}, \frac{\max\cos_j(z_k, W^j_{\text{Base}}) + \frac{1}{C-1}}{\max\cos_i(z_k, W^i_{\text{Incr}}) + \frac{1}{C-1} + \varepsilon}\right)\right), 1\right) \tag{4.5}$$

4.5.5 Comparison with other FSCIL works

Table 4.5 presents a comparison of our proposed method with other existing FSCIL approaches on the *mini*ImageNet benchmark. Among the methods compared, CEC [16], TEEN [21], SAVC [23], NC-FSCIL [25], and Yourself [24] share a common goal of enhancing **stability**, primarily by designing modules that operate during the **base stage** to improve representation learning. Our approach also belongs to this *stability-focused* category. Following the baseline NC-FSCIL [25], we adopt a lightweight CNN-based backbone, specifically ResNet-12, to maintain consistency in architecture and fair comparison. While several recent works employ more complex backbones such as ViT, we demonstrate that our method achieves competitive

TABLE 4.5: Comparison with other FSCIL works

Category	Alias	**Method**	**Stage**	**Backbone**	***mini*ImageNet**
–	*TOPIC [13]*	neural-gas structure	–	CNN	39.64
Stability	*CEC [16]*	attention module	Base	ResNet18	57.91
	TEEN [21]	Train free calibration	Post-hoc	ResNet18	61.76
	SAVC [23]	Pseudo-classes	Base	ResNet18	66.66
	NC-FSCIL [25]	ETF vector	Base	ResNet12	67.81
	Yourself [24]	Recitfy ViT feature	Base	ViT	68.80
	Ours	Uncertainty module + Calibration	Base	ResNet12	67.99
Adaptability	*+ subNet tuning [27]*	Binary Masking method	Incremental	ResNet18	60.86

performance even with a simple CNN backbone. Notably, our approach achieves the highest accuracy (67.99%) among all CNN-based methods on *mini*ImageNet, surpassing the NC-FSCIL baseline (67.81%).framework.

4.5.6 Limitations

Although our method achieves consistent improvements, it still has some limitations. First, due to the constraints of the FSCIL setting, the uncertainty distributions used for calibration are estimated from class prototypes and a limited number of few-shot samples, which does not fully capture the true uncertainty of the test data. Second, the uncertainty module occasionally exhibits unstable behavior during incremental sessions, sometimes collapsing to near-constant outputs. Future work should explore more robust training strategies and perform extensive evaluations under various hyperparameter settings. As discussed in Section 4.5.1, it is also necessary to develop a more accurate uncertainty estimation model that can better approximate the ground-truth distribution. Additionally, the obvious mismatch observed on the CUB200 dataset suggests that further investigation is required to address the challenges posed by smaller training sizes and potential overfitting or underfitting issues.

Chapter 5

Conclusion

5.1 Conclusion

In this work, we present an uncertainty-aware framework for Few-Shot Class-Incremental Learning (FSCIL), where a model must learn new classes from limited samples while retaining knowledge of previously learned ones. We propose a novel uncertainty score that quantifies the alignment between learned representations and class-specific ETF prototypes. This score is used to calibrate the model's prediction confidence in a class-aware manner, helping to reduce misclassifications of incremental samples into base classes. To estimate uncertainty distributions for calibration, we utilize memory prototypes for previous classes and few-shot data for current session classes, adhering to the constraints of the FSCIL setting. Extensive experiments on CIFAR100 and other benchmarks demonstrate that our approach improves overall model performance, primarily by achieving consistent gains in incremental class accuracy across several sessions, despite a slight drop in base class performance. Despite these improvements, our method has limitations. The estimated uncertainty distributions may not perfectly approximate the true test-time distribution, and we observe that the predicted uncertainty scores do not always align well with the supervised uncertainty loss signals. Additionally, when using the division-based uncertainty scoring function, the uncertainty module occasionally exhibits unstable behavior, such as collapsing to near-constant outputs.

Overall, our findings highlight the potential of integrating uncertainty estimation into FSCIL pipelines and offer a promising direction for more reliable and calibrated incremental learning.

Bibliography

[1] 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.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[3] G. Krempl, I. Žliobaite, D. Brzeziński, *et al.*, "Open challenges for data stream mining research," *ACM SIGKDD explorations newsletter*, vol. 16, no. 1, pp. 1–10, 2014.

[4] M. De Lange, R. Aljundi, M. Masana, *et al.*, "A continual learning survey: Defying forgetting in classification tasks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 7, pp. 3366–3385, 2021.

[5] 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.

[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] S. Mittal, S. Galesso, and T. Brox, "Essentials for class incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3513–3522.

[8] G. Kim, C. Xiao, T. Konishi, Z. Ke, and B. Liu, "A theoretical study on solving continual learning," *Advances in neural information processing systems*, vol. 35, pp. 5065–5079, 2022.

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

[10] 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.

[11] 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.

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

[13] 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. 12 183–12 192.

[14] 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.

[15] 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.

[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. 12 455–12 464.

[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] K. Zhu, Y. Cao, W. Zhai, J. Cheng, and Z.-J. Zha, "Self-promoted prototype refinement for few-shot class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6801–6810.

[19] 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*, vol. 35, 2021, pp. 1255–1263.

[20] S. Roy, C. Park, A. Fahrezi, and A. Etemad, "A bag of tricks for few-shot class-incremental learning," *arXiv preprint arXiv:2403.14392*, 2024.

[21] 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. 15 060–15 076, 2023.

[22] 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*, Springer, 2022, pp. 382–397.

[23] Z. Song, Y. Zhao, Y. Shi, P. Peng, L. Yuan, and Y. Tian, "Learning with fantasy: Semantic-aware virtual contrastive constraint for few-shot class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 24 183–24 192.

[24] Y.-M. Tang, Y.-X. Peng, J. Meng, and W.-S. Zheng, "Rethinking few-shot class-incremental learning: Learning from yourself," in *European Conference on Computer Vision*, Springer, 2024, pp. 108–128.

[25] 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.

[26] L. R. Marusich, J. Z. Bakdash, Y. Zhou, and M. Kantarcioglu, "Using ai uncertainty quantification to improve human decision-making," *arXiv preprint arXiv:2309.10852*, 2023.

[27] H. Kang, J. Yoon, S. R. H. Madjid, S. J. Hwang, and C. D. Yoo, "On the soft-subnetwork for few-shot class incremental learning," *arXiv preprint arXiv:2209.07529*, 2022.

[28] 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. 37 991–38 002, 2022.

[29] 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.

[30] 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. 27 225–27 238, 2022.

[31] T. Behnia, G. R. Kini, V. Vakilian, and C. Thrampoulidis, "On the implicit geometry of cross-entropy parameterizations for label-imbalanced data," in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2023, pp. 10 815–10 838.

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

[33] X. Li, S. Liu, J. Zhou, *et al.*, “Principled and efficient transfer learning of deep models via neural collapse,” *arXiv e-prints*, arXiv–2212, 2022.

[34] M. Kirchhof, B. Mucsányi, S. J. Oh, and D. E. Kasneci, “Url: A representation learning benchmark for transferable uncertainty estimates,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 13 956–13 980, 2023.

[35] D. Tran, J. Liu, M. W. Dusenberry, *et al.*, “Plex: Towards reliability using pretrained large model extensions,” *arXiv preprint arXiv:2207.07411*, 2022.

[36] K. Zou, Z. Chen, X. Yuan, X. Shen, M. Wang, and H. Fu, “A review of uncertainty estimation and its application in medical imaging,” *Meta-Radiology*, vol. 1, no. 1, p. 100 003, 2023.

[37] M.-H. Laves, S. Ihler, J. F. Fast, L. A. Kahrs, and T. Ortmaier, “Well-calibrated regression uncertainty in medical imaging with deep learning,” in *Medical imaging with deep learning*, PMLR, 2020, pp. 393–412.

[38] A. Krizhevsky, G. Hinton, *et al.*, “Learning multiple layers of features from tiny images,” 2009.

[39] O. Russakovsky, J. Deng, H. Su, *et al.*, “Imagenet large scale visual recognition challenge,” *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[40] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” 2011.