Continual Out-of-Distribution Detection with Analytic Neural Collapse

Saleh Momeni[1], **Changnan Xiao**[2], **Bing Liu**[1]

[1] Department of Computer Science, University of Illinois Chicago
[2] ChangnXX.github.io
{smomen3, liub}@uic.edu changnanxiao@gmail.com

Abstract

Continual learning (CL) aims to enable models to incrementally learn from a sequence of tasks without forgetting previously acquired knowledge. While most prior work focuses on closed-world settings, where all test instances are assumed from the set of learned classes, real-world applications require models to handle both CL and out-of-distribution (OOD) samples. A key insight from recent studies on deep neural networks is the phenomenon of Neural Collapse (NC), which occurs in the terminal phase of training when the loss approaches zero. Under NC, class features collapse to their means, and classifier weights align with these means, enabling effective prototype-based strategies such as *nearest class mean*, for both classification and OOD detection. However, in CL, catastrophic forgetting (CF) prevents the model from naturally reaching this desirable regime. In this paper, we propose a novel method called Analytic Neural Collapse (AnaNC) that analytically creates the NC properties in the feature space of a frozen pre-trained model with no training, overcoming CF. Extensive experiments demonstrate that our approach outperforms state-of-the-art methods in continual OOD detection and learning, highlighting the effectiveness of our method in this challenging scenario.

Code — https://github.com/salehmomeni/AnaNC

Introduction

Continual learning (CL) aims to equip models with the ability to incrementally learn a series of tasks arriving over time. Each task typically introduces new classes, which the model must learn without access to data from previous tasks and classify without knowing the task identity at test time. This setting, known as *class-incremental learning* (CIL) (Van de Ven and Tolias 2019), assumes all test instances belong to the set of learned classes. However, many real-world applications require the continual model not only to classify among known classes but also to recognize when an input belongs to an unseen class, known as *out-of-distribution* (OOD) detection (Fei and Liu 2016; Hendrycks and Gimpel 2017). In this work, we address this problem in the CIL setting, performing *continual OOD detection*, where the model must detect OOD samples throughout the learning process.

A central issue in CL, whether for classification or OOD detection, is *catastrophic forgetting* (CF), where learning new tasks causes the model to overwrite representations of previously learned tasks (McCloskey and Cohen 1989). A promising strategy to overcome CF is to use pre-trained models (PTMs), which provide generalizable representations across tasks. Using PTMs reduces the risk of CF and yields strong CL performance (Ke et al. 2021; Zhou et al. 2024a; Momeni, Mazumder, and Liu 2025).

Deep neural networks trained for classification often exhibit a phenomenon known as *Neural Collapse* (NC) in the terminal phase of training (TPT). During this phase, the geometry of the learned features and classifier weights evolves toward a highly structured form (Papyan, Han, and Donoho 2020), characterized by the following properties:

- **NC1 (Collapse of Within-class Variability):** the within-class variation becomes negligible as each feature representation collapses toward its respective class mean.
- **NC2 (Equiangular Tight Frame of Class Means):** the class means have equal norms and are uniformly separated by equal angles, forming a symmetric configuration known as an Equiangular Tight Frame (ETF), which maximizes the inter-class separation.
- **NC3 (Self-duality Between Features and Classifier):** the weights of the network's linear classifier become aligned with the class means, meaning that the decision boundaries become geometrically similar, with each class mean centered at its corresponding decision region.
- **NC4 (Classifier Reduction to Nearest Class Mean):** the network's prediction reduces to the *nearest class mean* (NCM) rule, where a test sample is assigned the label of the class whose mean is the closest, typically measured using Euclidean distance.

In practice, NC1 and NC2 are sufficient to infer all the properties: when all features within a class are identical and the class means form a simplex ETF, then the optimal classifier must align its weights with those class means. This structure has been shown to improve the performance of OOD detection (Haas, Yolland, and Rabus 2023; Ammar et al. 2023).

NC4 provides an alternative to training a linear classifier, which is prone to CF, by enabling the use of the NCM strategy for both classification and OOD detection. While NCM

has been adopted in CL, prior work has overlooked the underlying NC insights and applied it without realizing NC properties (Zhou et al. 2024b). The NC phenomenon typically arises only in the terminal phase of training, i.e., the network has been extensively optimized and the training loss approaches zero (Papyan, Han, and Donoho 2020). However, achieving NC in CL is difficult because incremental training causes CF, leading to degradation of earlier class representations. Prior work has shown that the NC geometry fails to emerge under such conditions (Yang et al. 2023).

To address this challenge, we introduce Analytic Neural Collapse (AnaNC), a novel method that constructs the NC geometry in closed-form within the feature space of a frozen PTM. AnaNC applies an analytic projection that realizes NC1 and NC2, producing a geometry similar to what emerges in fully trained networks but without any optimization, thereby avoiding CF. This NC geometry then supports distance-based continual learning and OOD detection using class means in accordance with NC4.

Related Work

OOD detection has been extensively studied in the literature, often under related terms such as *outlier detection* or *anomaly detection* (Malinin and Gales 2019; Ghassemi and Fazl-Ersi 2022). Existing methods can be divided into several groups: The first group leverages the classifier's output to derive OOD scores. A widely used baseline in this category is the Maximum Softmax Probability (Hendrycks and Gimpel 2017), which uses the highest predicted class probability as a confidence measure. Subsequent works have proposed alternatives such as MaxLogit, KL divergence matching (Hendrycks et al. 2022), or energy-based scores (Liu et al. 2020; Wang et al. 2021) to better capture uncertainty at the classifier's output.

Various approaches focus on enhancing OOD detection by manipulating the features, using techniques such as clipping (Sun, Guo, and Li 2021), sparsification (Sun and Li 2022), or input perturbations (Liang, Li, and Srikant 2018) to better separate in- and out-of-distribution samples.

Another category of methods exploits the geometry of feature space by projecting samples onto the principal or null space of ID classes and measuring deviations from these low-dimensional manifolds (Cook, Zare, and Gader 2020; Wang et al. 2022b; Ammar et al. 2023).

The final category relies on distance metrics, such as Mahalanobis distance (Lee et al. 2018; Ren et al. 2021) or nearest-neighbor (Sun et al. 2022), to compute a sample's distance to known classes or training points. Our method falls within this category but targets continual OOD detection (Aljundi et al. 2022), where classes arrive incrementally.

Continual OOD detection has also been studied, though most OOD methods are not directly applicable here because they assume access to all data at once. Some studies have adapted CL techniques for this setting. For example, Aguilar et al. (2023) combined knowledge distillation with uncertainty estimation to mitigate CF, while Gummadi et al. (2022) leveraged high-level features for OOD detection and low-level features with regularization to accommodate new classes. An early work by Bendale and Boult (2015) introduced the nearest non-outlier, which utilizes NCM with a linear transformation, but it does not realize any NC properties. Kaymak et al. (2025) proposed a continual OOD detection system that leverages Mahalanobis distance. Miao et al. (2025) conducted a comprehensive benchmark study, while He and Zhu (2022) investigated OOD detection in the context of unsupervised CL. Kim et al. (2025) presented a theoretical analysis of OOD detection in CL.

Continual OOD detection in *task-incremental learning* (TIL) is also studied (Kim et al. 2022; Rios et al. 2022; Liu, Zhao, and Guo 2025), where a separate model is trained for each task. This contrasts with our focus in this paper on CIL setting, where a single model must handle all tasks.

Continual Learning mainly focused on addressing the CF issue (Wang et al. 2024). Many methods utilize regularization to penalize changes to parameters deemed important for prior tasks, thereby helping to preserve earlier knowledge (Kirkpatrick et al. 2017; Rebuffi et al. 2017). Knowledge distillation is another strategy, where the output of the previous model is used to guide the learning of the current model (Li and Hoiem 2017; Buzzega et al. 2020). Another widely used method is experience replay, where a subset of samples from previous tasks is stored in a memory buffer and used during training to help retain performance on earlier tasks (Aljundi et al. 2019; Chaudhry et al. 2019; Wang et al. 2022a). Parameter isolation offers a different solution by separating knowledge of different tasks using techniques such as masking (Serra et al. 2018; Wortsman et al. 2020) or orthogonal projection (Zeng et al. 2019).

The use of PTMs has become increasingly common in CL and has resulted in significant accuracy gains (Zhou et al. 2024a). One line of work focuses on learning prompts to guide predictions while keeping the PTM frozen to preserve its acquired knowledge (Wang et al. 2022c; Smith et al. 2023; Wang et al. 2023; Roy et al. 2024). Other approaches explore fine-tuning strategies, either by updating the full model (Zhang et al. 2023) or by introducing lightweight adapters (Liang and Li 2024; Sun et al. 2025). Prototype-based methods are also popular in CL, representing each class by a mean vector (Zhou et al. 2024b) or a Gaussian distribution (Hayes and Kanan 2020; Goswami et al. 2023; Momeni, Mazumder, and Liu 2025). Several works cast CL as a regression problem (Zhuang et al. 2022; McDonnell et al. 2023; Peng et al. 2025). We likewise employ ridge regression, but unlike these methods that predict one-hot labels, our approach induces the NC geometry.

Problem Definition

Continual learning aims to incrementally acquire knowledge from a sequence of tasks, typically involving previously unseen classes. Formally, at each task t, the model receives a training dataset $\mathcal{D}_t = \{(x_t^{(i)}, y_t^{(i)})\}_{i=1}^{n_t}$, where $x_t^{(i)} \in \mathcal{X}_t$ is an input sample, and $y_t^{(i)} \in \mathcal{Y}_t$ is the corresponding label.

In *class-incremental learning* setting of CL, the class sets $\mathcal{Y}_t$ are disjoint across tasks, and the goal is to learn a single model capable of making predictions without knowing the task identity at test time.

In this work, we target the CIL setting and continual OOD detection, as models in real-world applications must inevitably handle samples outside the learned distribution. We place particular emphasis on continual OOD detection in our experiments, as it is less explored in the literature.

Preliminaries

Ridge Regression: Let $X \in \mathbb{R}^{N\times d}$ denote a set of input features extracted from a PTM. The objective is to map the input features to a target matrix $Y \in \mathbb{R}^{N\times d}$. Here, we use d to represent both the input and output dimensions for simplicity, although they may differ in general. Ridge regression finds the optimal linear mapping by minimizing the squared error with L_2 regularization:

$$\min_{W} \|XW - Y\|^2 + \lambda\|W\|^2 \tag{1}$$

where $\lambda > 0$ controls the degree of regularization, discouraging large weight magnitudes to improve generalization. The solution to this problem is obtained analytically by setting the gradient with respect to W to zero, leading to the expression:

$$W = (X^\top X + \lambda I)^{-1} X^\top Y \tag{2}$$

known as the ridge regression or regularized least squares. In this expression, $X^\top X$ is commonly referred to as the *Gram matrix*, and we call $X^\top Y$ the cross matrix.

In CL, where tasks arrive sequentially and storing past data is infeasible, it is possible to update the ridge regression solution incrementally (Liang et al. 2006; Zhuang et al. 2022; McDonnell et al. 2023). Given new task data (X_t, Y_t), the gram and cross matrices can be updated recursively:

$$G_t = G_{t-1} + X_t^\top X_t, \quad H_t = H_{t-1} + X_t^\top Y_t \tag{3}$$

where G_t and H_t are the gram and cross matrices after task t, respectively. This allows us to compute the updated solution:

$$W_t = (G_t + \lambda I)^{-1} H_t \tag{4}$$

This formulation allows for incremental updates without revisiting old data, which is required by CL.

Extreme Learning Machines: To enhance feature expressiveness, the input dimension d can be randomly projected into a higher-dimensional space of dimension D before learning the linear regression:

$$Z = \phi(XW_{rp}) \tag{5}$$

where $W_{rp} \in \mathbb{R}^{d\times D}$ is a fixed matrix with random values and $\phi(\cdot)$ is a nonlinear activation, for which we use a GELU function. This transformation helps capture nonlinear interactions between input features, that may not be adequately represented in the original space (Huang et al. 2011). Without loss of generality, other non-linear transformations such as kernel functions may also be employed to achieve a similar effect (Momeni, Mazumder, and Liu 2025). The mapping to output targets can be learned as before:

$$W = (Z^\top Z + \lambda I)^{-1} Z^\top Y \tag{6}$$

This setup follows the Extreme Learning Machine (ELM) framework (Huang et al. 2011; Liang et al. 2006), where a single-layer feedforward network is constructed using randomly initialized hidden layer weights. The key idea behind ELMs is to separate representation learning (driven by the random projection and nonlinear activation) from the optimization, allowing the output layer weights to be derived analytically. Note that this solution can be updated incrementally using the recursive updates introduced in Equations 3 and 4.

Methodology

The core idea behind our approach is to leverage class means as prototypes for prediction, following the insights from NC. In particular, NC4 suggests that the class means and the classifier weights align, enabling a simple yet effective NCM classification. However, directly applying this strategy in CL, without realizing the geometric properties associated with NC will lead to poor performance. This is because NC only emerges in the terminal phase of training when the network is trained on all classes with near-zero classification loss, a condition that is not feasible in CL due to the incremental arrival of tasks and the challenge of CF.

To overcome this, we introduce AnaNC, a novel projection-based method designed to achieve the geometric properties of NC without requiring training of the PTM. Our approach is inspired by the principles of ELMs, which provide an efficient analytic solution.

The overall pipeline of our method is illustrated in Figure 1 (left). We begin by using a frozen PTM to extract features from input samples, which forms the **input layer** of our system. To improve expressiveness, these features are then projected into a higher-dimensional space, referred to as the **random projection (RP) layer**, where random weights and nonlinear activations are applied. The transformed features are then mapped to the output space denoted as the **output layer**, which is designed to satisfy the geometric properties of NC. The effect of this projection on the feature geometry is illustrated in Figure 1 (right), showing how it creates the structured, symmetric arrangement of NC. In the following sections, we detail the construction of this projection and how the entire system operates incrementally for CL.

Achieving Class-variation Collapse (NC1)

We can leverage the ELM framework to map input features $X \in \mathbb{R}^{N\times d}$ to a target matrix $Y \in \mathbb{R}^{N\times d}$ that defines the desired structure of the output feature space. The ELM solution can be written in a compact form as:

$$W = (Z^\top Z + \lambda I)^{-1} H \tag{7}$$

where $Z \in \mathbb{R}^{N\times D}$ represents the random features produced by the RP layer, and $H = Z^\top Y$ is the cross matrix between the random features and the target outputs.

To collapse within-class variability and achieve NC1, all samples belonging to a class c should be mapped to the same point in the output space, denoted as $\tilde{\mu}_c \in \mathbb{R}^d$. This target point doesn't need to match the original class mean from the PTM features, but instead will become the new class mean.

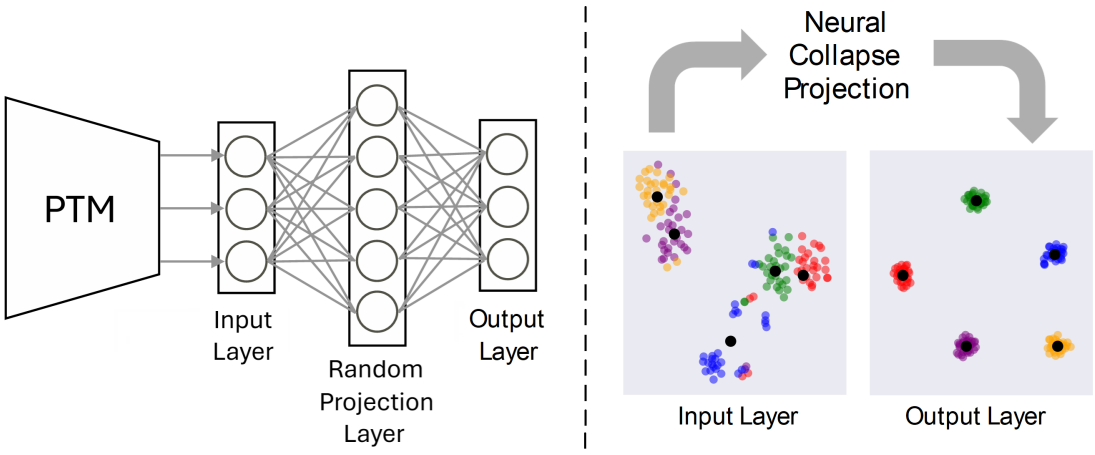

Figure 1: (**Left**) Overview of the proposed AnaNC architecture. The random projection layer applies fixed, randomly initialized weights to enhance feature expressiveness, while the output layer realizes NC properties via an analytic solution. (**Right**) t-SNE visualization of input features (five classes of CUB dataset) and their arrangements after AnaNC using DINO as the PTM.

Let $\tilde{M} = [\tilde{\mu}_1, \tilde{\mu}_2, \ldots, \tilde{\mu}_C]$ be the matrix of target means for all C classes learned so far, the cross matrix can be expanded as:

$$H = Z^\top Y = \sum_c \left(\sum_{i \in c} z_i \right) \tilde{\mu}_c^\top = \sum_c r_c n_c \tilde{\mu}_c^\top = RN\tilde{M} \tag{8}$$

where $r_c \in \mathbb{R}^D$ is the mean of the random features for class c, $R \in \mathbb{R}^{D \times C}$ is the matrix containing all the random feature means, and $N = \text{diag}(n_1, \ldots, n_C)$ is a diagonal matrix containing the class sample counts.

This formulation enables incremental updates by maintaining the matrix R of the random feature means, class sample counts N, and providing the target means $\tilde{M}$, which we will explain in the following section. The learned projection W thus maps all the feature representations of a class to their corresponding target mean $\tilde{\mu}_c$, collapsing the inter-class variation in the output space and satisfying NC1.

Achieving Simplex ETF (NC2)

To achieve NC2, we aim to arrange the class means into an ETF, a symmetric configuration that ensures equal pairwise inner products between all class means (Papyan, Han, and Donoho 2020). Formally, the centered and normalized target means should satisfy:

$$\langle \tilde{\mu}_i, \tilde{\mu}_j \rangle = \begin{cases} 1 & \text{if } i = j \\ -\frac{1}{C-1} & \text{if } i \neq j \end{cases} \tag{9}$$

This results in a product matrix $S \in \mathbb{R}^{C \times C}$ defined as:

$$S = I - \frac{1}{C-1}(\mathbf{1}\mathbf{1}^\top - I) \tag{10}$$

Geometrically, this corresponds to a regular simplex inscribed on the unit hypersphere in $\mathbb{R}^{C-1}$, where the means are uniformly distributed. Importantly, directly assigning the original class means to arbitrary ETF vertices can make the mapping difficult to learn, as it may introduce large transformation in class positions and require a complete reorganization of the feature space. Motivated by this, we aim to find target means $\tilde{M}$ that satisfies the ETF constraint $\tilde{M}\tilde{M}^\top = S$, while remaining as close as possible (in Frobenius norm) to the original class means M. Formally, we solve the following constrained optimization problem:

$$\min_{\tilde{M} \in \mathbb{R}^{C \times D}} \|\tilde{M} - M\|_F^2 \quad \text{subject to} \quad \tilde{M}\tilde{M}^\top = S \tag{11}$$

Note that S is symmetric and positive semidefinite (PSD), so it has an eigen-decomposition: $S = U_S \Lambda U_S^\top$ where $U_S \in \mathbb{R}^{C \times C}$ is orthonormal and $\Lambda \in \mathbb{R}^{C \times C}$ is diagonal with non-negative eigenvalues. Let us define:

$$\tilde{M} = U_S \Lambda^{1/2} Q^\top \tag{12}$$

where $Q \in \mathbb{R}^{D \times C}$ is a semi-orthogonal matrix with $Q^\top Q = I$. This construction ensures that any matrix of this form satisfies the ETF constraint:

$$\tilde{M}\tilde{M}^\top = U_S \Lambda^{1/2} Q^\top Q \Lambda^{1/2} U_S^\top = S \tag{13}$$

We now must choose Q such that the resulting $\tilde{M}$ is as close as possible to M, i.e., we minimize:

$$\min_{Q^\top Q = I} \|U_S \Lambda^{1/2} Q^\top - M\|_F^2 \tag{14}$$

This is a well-established optimization known as the orthogonal Procrustes problem (Gower and Dijksterhuis 2004), and it admits a closed-form solution using the following lemma:

Lemma 1 (Orthogonal Procrustes Problem). *Let $A \in \mathbb{R}^{C \times C}$ and $B \in \mathbb{R}^{C \times D}$, with $C \leq D$. The solution to the optimization problem:*

$$\min_{Q \in \mathbb{R}^{D \times C}} \left\| AQ^\top - B \right\|_F^2 \quad \textit{subject to} \quad Q^\top Q = I$$

is given by

$$Q^\star = UV^\top$$

where $B^\top A = U \Sigma V^\top$ is the singular value decomposition (SVD) of $B^\top A$.

In our case, denote $A = U_S\Lambda^{1/2}$ and $B = M$. Applying the lemma, we compute the SVD of $M^\top U_S\Lambda^{1/2} = U\Sigma V^\top$ and set $Q^\star = UV^\top$. The closest matrix to M satisfying $\tilde{M}\tilde{M}^\top = S$ is then given by:

$$\tilde{M} = AQ^{\star\top} = U_S\Lambda^{1/2}VU^\top \tag{15}$$

This yields target means that follow the ETF structure, thereby achieving the NC2 condition. When a new task arrives, $\tilde{M}$ is recomputed to incorporate the new target means. The updated $\tilde{M}$ is used to form the cross matrix in Equation 8, and obtain the output layer weights W via Equation 7. Passing the PTM features through the ELM with these weights aligns the feature representations with the NC geometry. Therefore, **classification and OOD detection** can be performed by computing distances to the target means, following the NCM rule implied by NC4. We use cosine similarity as the distance metric, which is equivalent to Euclidean distance on normalized features.

Empirical Evaluation

Datasets: We evaluate our method on four publicly available image classification datasets commonly used in CL: CIFAR-100 (100 classes) (Krizhevsky, Hinton et al. 2009), ImageNet-R (200 classes) (Hendrycks et al. 2021), CUB (200 classes) (Wah et al. 2011), and Stanford Cars (196 classes) (Yang et al. 2015). For each dataset, we randomly shuffle the classes and partition them into 10 disjoint tasks. To account for variation in class-to-task assignment, we repeat all experiments using three different random seeds.

As mentioned earlier, our main focus is on continual OOD detection. Nonetheless, we also show strong performance on CIL. For OOD detection, we consider two settings:

- **In-dataset OOD:** At each task, test samples from future (unseen) tasks are treated as OOD. This is a challenging setting, as OOD samples come from the same dataset. We consider this ***our primary setting*** as it is more likely to be the case in practice.
- **Cross-dataset OOD:** The OOD data is sourced from another dataset. This is ***our secondary setting*** to evaluate the robustness of the proposed method. We use CIFAR-100 as the ID dataset and, following (Yang et al. 2022), include both near OOD datasets (CIFAR-10 (Krizhevsky, Hinton et al. 2009), Tiny ImageNet (Le and Yang 2015)) and far OOD datasets (Places365 (Zhou et al. 2017), FashionMNIST (Xiao, Rasul, and Vollgraf 2017)) to assess robustness under varying distribution shifts.

In all experiments, the model operates under the CIL setting without access to task identity or any replay data.

Baselines: We compare against OOD detection approaches that can be adapted for a continual setting and state-of-the-art CIL methods. Specifically, we include NCM, Mahalanobis Distance (MD) (Lee et al. 2018), NECO (Ammar et al. 2023), Residuals [1], KLDA (Momeni, Mazumder, and Liu 2025), FECAM (Goswami et al. 2023), CODA-Prompt (Smith et al. 2023), SLCA (Zhang et al. 2023), and RanPac (McDonnell et al. 2023). For CIL baselines that do not provide a dedicated OOD detection mechanism, we use their maximum logit as the OOD indicator.

For all methods that utilize the PTM only as a feature extractor, we adopt First-section Adaptation (FSA) following (Zhou et al. 2024b), by adding adapters to the backbone and fine-tuning on the first task to improve initial representation quality. This strategy is applied to all baselines except CODA-Prompt, SLCA, and RanPac, as CODA-Prompt employs prompt learning, SLCA incrementally fine-tunes the PTM, and RanPac incorporates its own FSA mechanism.

Implementation Details: For our main experiments, we use two self-supervised PTMs, DINO (Caron et al. 2021) and MOCO (Chen, Xie, and He 2021), trained on ImageNet-1K (Deng et al. 2009). We opted for self-supervised PTMs to avoid information leakage, as supervised models are exposed to class labels during pre-training, some of which may reappear during CL, giving the model prior knowledge. For ablation studies, we use the stronger DINO backbone.

We set the RP dimension to 5000 by default; we also include an ablation study to assess the impact of different RP dimensions on performance. The regularization parameter λ in the ELM is empirically set to 10^2 by searching over the range $[10^{-2}, 10^6]$ with a multiplicative factor of 10. This value works well for all datasets and PTMs. All experiments are run on an NVIDIA A100 GPU with 80GB of VRAM.

Evaluation Metrics: Identifying OOD samples requires setting a threshold on the OOD score. However, selecting an appropriate threshold is not the focus of this work, as it depends on the application. Therefore, we report two standard threshold-independent metrics: Area Under the ROC Curve (AUC), which quantifies the overall separability between ID and OOD samples across all thresholds, and FPR95, which measures the proportion of ID samples mistakenly classified as OOD when the model correctly identifies 95% of OOD samples. For both metrics, we compute values after each task is learned and report their average over all tasks.

For CIL classification performance, we use two standard metrics: Last Accuracy (A_{last}), which is the final classification accuracy after all tasks have been learned, and Average Incremental Accuracy (A_{avg}), which averages the classification accuracy measured after each task.

Continual OOD Detection Results

In-dataset OOD Setting. The results of our primary continual OOD detection setting are given in Table 1, where samples from future unseen tasks are considered OOD. AnaNC outperforms all baselines with both PTMs overall. Using DINO, we observe an average improvement of 2.22% in AUC and 3.24% in FPR95 over the best baseline. With MOCO, the gains are 1.51% in AUC and 4.38% in FPR95.

Effect of RP Dimension: We use a default RP dimension of $D = 5000$ in our main experiments. To analyze its impact, we evaluate the effect of varying D on the in-dataset OOD detection setting as shown in Figure 2. Increasing D

[1]Residuals is proposed in ViM (Wang et al. 2022b), where it is integrated with logits for OOD detection. While residuals can be adapted for CL, combining it with logits is infeasible in this setting.

	Method	CIFAR100		ImageNet-R		CUB		Cars		Average	
		AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓
DINO ImageNet-1K	NCM	84.63 ± 1.98	60.36 ± 3.33	74.14 ± 0.59	83.75 ± 0.83	71.18 ± 1.22	84.78 ± 1.46	58.68 ± 0.32	92.80 ± 0.61	72.16	80.42
	MD	82.50 ± 1.80	63.91 ± 3.98	71.69 ± 0.79	83.18 ± 1.15	64.28 ± 1.17	89.83 ± 0.65	60.48 ± 0.38	92.69 ± 0.20	69.74	82.40
	KLDA	85.25 ± 1.56	56.40 ± 3.77	75.08 ± 1.15	**79.08** ± 0.34	71.33 ± 1.43	83.55 ± 1.44	60.34 ± 0.97	92.01 ± 0.62	73.00	77.76
	FECAM	84.33 ± 1.28	62.79 ± 2.65	73.34 ± 0.26	83.67 ± 0.88	73.26 ± 1.84	81.44 ± 2.16	67.79 ± 0.74	89.89 ± 0.72	74.68	79.44
	Residuals	75.21 ± 1.58	74.47 ± 3.00	65.75 ± 1.00	88.66 ± 0.82	59.62 ± 0.79	90.92 ± 0.26	58.13 ± 0.13	93.33 ± 0.11	64.68	86.84
	NECO	74.87 ± 1.07	75.04 ± 2.33	65.53 ± 0.77	89.21 ± 0.97	59.86 ± 1.21	90.98 ± 0.52	58.05 ± 0.20	92.98 ± 0.17	64.58	87.05
	CODA-P	66.46 ± 0.48	74.56 ± 1.16	60.54 ± 1.27	90.66 ± 1.56	61.40 ± 3.56	85.04 ± 3.58	52.91 ± 1.19	92.91 ± 0.33	60.33	85.79
	SLCA	85.10 ± 1.22	64.29 ± 1.73	71.73 ± 1.95	88.11 ± 1.13	73.57 ± 2.84	76.89 ± 1.91	74.30 ± 0.12	79.00 ± 0.61	76.17	77.07
	RanPac	86.63 ± 0.23	**55.12** ± 1.27	77.84 ± 0.32	82.58 ± 0.85	80.64 ± 0.49	73.94 ± 1.52	76.63 ± 0.60	79.88 ± 1.49	80.44	72.88
	AnaNC (ours)	**86.94** ± 1.07	55.17 ± 2.70	**78.87** ± 0.40	80.80 ± 0.73	**83.07** ± 0.53	**66.45** ± 1.93	**81.76** ± 0.80	**76.13** ± 1.01	**82.66**	**69.64**
MOCO ImageNet-1K	NCM	76.22 ± 2.52	68.21 ± 2.74	73.81 ± 0.13	80.44 ± 0.43	72.22 ± 2.41	78.89 ± 2.72	57.01 ± 1.56	92.23 ± 0.28	69.81	79.94
	MD	79.16 ± 1.94	60.69 ± 1.01	73.33 ± 0.56	76.97 ± 0.81	66.23 ± 1.54	87.42 ± 0.76	58.05 ± 1.47	92.62 ± 0.76	69.19	79.42
	KLDA	78.90 ± 2.49	61.26 ± 1.21	73.78 ± 0.12	**76.02** ± 0.90	68.94 ± 2.12	83.69 ± 1.51	58.78 ± 1.49	91.65 ± 0.68	70.10	78.15
	FECAM	79.55 ± 2.42	62.26 ± 1.11	74.79 ± 0.62	78.41 ± 0.95	73.66 ± 2.31	76.46 ± 3.10	64.02 ± 2.01	88.34 ± 0.83	73.00	76.36
	Residuals	77.80 ± 2.28	64.16 ± 1.48	70.69 ± 0.84	80.12 ± 0.66	61.91 ± 1.70	89.37 ± 1.15	56.75 ± 1.76	92.77 ± 0.18	66.79	81.61
	NECO	76.41 ± 1.89	66.11 ± 0.71	71.46 ± 0.10	79.58 ± 0.84	61.96 ± 1.49	89.16 ± 0.67	56.73 ± 1.55	92.84 ± 0.31	66.64	81.92
	CODA-P	54.57 ± 0.23	86.77 ± 0.10	54.54 ± 0.38	92.90 ± 0.68	55.04 ± 1.76	91.53 ± 0.59	48.31 ± 0.56	94.78 ± 0.05	53.12	91.50
	SLCA	79.63 ± 0.58	75.51 ± 0.83	65.71 ± 0.20	91.17 ± 1.15	68.26 ± 2.83	85.28 ± 1.64	64.38 ± 0.50	87.62 ± 1.14	69.49	84.90
	RanPac	83.84 ± 1.49	64.36 ± 3.03	76.21 ± 0.02	83.91 ± 0.27	79.31 ± 0.84	76.88 ± 1.71	78.70 ± 0.74	79.52 ± 1.36	79.52	76.17
	AnaNC (ours)	**84.21** ± 1.60	**59.47** ± 2.10	**77.96** ± 0.43	81.16 ± 0.52	**81.87** ± 0.84	**67.83** ± 1.88	**80.07** ± 1.69	**78.71** ± 2.03	**81.03**	**71.79**

Table 1: OOD detection performance under the in-dataset setting. Results are averaged over three random class-task splits, with standard deviation reported.

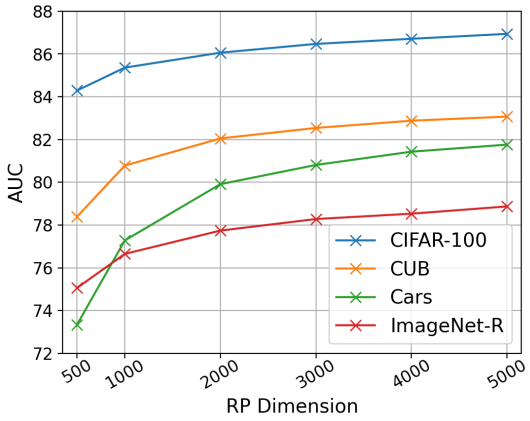

Figure 2: OOD detection performance (AUC) with DINO backbone under the in-dataset setting for varying number of RP dimensions.

improves OOD detection performance across all datasets by enhancing the expressiveness of the RP layer and enabling richer feature representations.

While higher dimensions increase the size of the RP matrix $W_{\text{RP}} \in \mathbb{R}^{d\times D}$, the output layer weights $W \in \mathbb{R}^{D\times d}$, and the $D \times D$ gram matrix, we note that these parameters are non-trainable. As such, the additional computational cost is minimal, effectively resembling the addition of a feedforward layer at the end of the PTM.

Analysis of Analytic Neural Collapse: To understand how our method creates the NC properties, we evaluate OOD detection performance of NCM with four sets of features: (1) the original PTM feature, (2) the features of RP layer, (3) output layer features when NC1 is applied by mapping each sample to its class mean, and (4) output layer feature when both NC1 and NC2 are applied by arranging the class means on an ETF simplex and mapping samples to these target means before.

Features	CIFAR100		ImageNet-R		CUB		Cars	
	AUC	FPR95	AUC	FPR95	AUC	FPR95	AUC	FPR95
Input	84.63	60.36	74.14	83.75	71.18	84.78	58.68	92.80
RP	85.09	59.18	74.51	82.40	70.61	84.71	59.14	92.58
NC1	86.30	58.88	76.34	81.84	78.56	72.39	69.52	87.61
NC1&2	**86.94**	**55.17**	**78.87**	**80.80**	**83.07**	**66.45**	**81.76**	**76.13**

Table 2: Ablation study of AnaNC using a DINO backbone under the in-dataset OOD setting. We report NCM performance using: the original PTM features at the input layer, RP features with dimension 5000, output layer features after creating only NC1, and output layer features after creating both NC1 and NC2 (proposed method).

The results in Table 2 indicate that applying RP alone is insufficient for improving NCM performance, as its primary role is to expand the feature space for ELM. Enforcing NC1 significantly enhances OOD detection performance, and incorporating NC2 leads to additional improvement. This highlights the effectiveness of our method in achieving both NC properties and inducing the desired geometry.

Cross-dataset Setting. The continual OOD detection results for this setting are given in Table 3, where we use CIFAR-100 as the ID dataset and assess the model's ability to detect OOD samples from four other datasets.

AnaNC again achieves the best overall performance, improving DINO results by 0.49% in AUC and 0.88% in FPR95, and MOCO results by 0.86% in AUC and 1.94% in FPR95 on average compared to the strongest baselines. Here, the improvements are smaller since these OOD classes are more distant and thus less challenging to detect.

Memory and Running Time Efficiency. The proposed AnaNC introduces minimal memory overhead. Specifically, it adds a feedforward layer with input dimension d and intermediate dimension D to the PTM. Additionally, AnaNC maintains a $D \times D$ gram matrix, class means $\mu_c \in \mathbb{R}^d$, and random feature means $r_c \in \mathbb{R}^D$. For a typical setting of $d = 768$, $D = 5000$, and $C = 100$, the total parame-

	Method	CIFAR10		T-ImageNet		Places365		FashionMNIST		Average	
		AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓	AUC ↑	FPR95 ↓
DINO ImageNet-1K	NCM	85.39 ± 3.10	52.09 ± 4.46	89.06 ± 0.15	46.55 ± 0.84	87.35 ± 1.04	50.94 ± 2.02	96.29 ± 1.14	20.02 ± 7.70	89.52	42.40
	MD	80.83 ± 2.84	65.57 ± 5.93	85.89 ± 0.96	57.16 ± 3.43	**89.59** ± 2.13	49.26 ± 6.57	95.49 ± 1.53	28.87 ± 11.2	87.95	50.21
	KLDA	86.12 ± 2.82	49.46 ± 5.58	88.14 ± 0.95	50.34 ± 2.99	87.53 ± 2.18	**48.49** ± 4.33	96.51 ± 0.58	19.75 ± 3.04	89.57	42.01
	FECAM	83.52 ± 2.90	61.01 ± 4.60	89.32 ± 0.32	47.81 ± 1.47	88.11 ± 1.26	49.70 ± 4.12	**97.15** ± 0.18	**16.54** ± 1.51	89.52	43.76
	Residuals	70.84 ± 3.45	80.33 ± 4.36	79.78 ± 0.14	70.39 ± 0.58	86.84 ± 2.04	58.37 ± 5.36	90.59 ± 1.82	54.65 ± 8.55	82.01	65.94
	NECO	70.36 ± 3.44	81.50 ± 4.28	78.61 ± 0.60	73.62 ± 2.32	87.05 ± 1.46	59.15 ± 4.11	88.50 ± 1.99	62.96 ± 7.60	81.13	69.31
	CODA-P	86.01 ± 2.09	55.96 ± 6.72	88.18 ± 0.27	48.40 ± 1.36	87.57 ± 0.26	51.92 ± 3.49	96.11 ± 0.11	20.34 ± 0.94	89.47	44.16
	SLCA	86.79 ± 0.94	55.26 ± 1.57	88.48 ± 0.12	48.63 ± 1.56	88.07 ± 0.17	56.33 ± 0.73	96.00 ± 0.30	25.25 ± 0.80	89.84	46.37
	RanPac	87.83 ± 2.19	48.55 ± 5.19	88.98 ± 0.05	45.85 ± 1.55	87.57 ± 0.76	52.00 ± 0.54	96.33 ± 0.21	20.08 ± 1.19	90.18	41.62
	AnaNC (ours)	**88.18** ± 2.08	**48.31** ± 4.42	**89.54** ± 0.11	**44.53** ± 1.18	88.21 ± 0.59	50.82 ± 0.53	96.77 ± 0.23	19.31 ± 2.04	**90.67**	**40.74**
MOCO ImageNet-1K	NCM	76.15 ± 2.86	62.00 ± 7.62	79.58 ± 1.37	63.59 ± 4.20	74.74 ± 2.62	74.35 ± 3.73	91.37 ± 3.30	41.69 ± 16.5	80.46	60.41
	MD	77.51 ± 3.63	58.44 ± 4.56	81.26 ± 1.02	58.80 ± 1.86	78.97 ± 1.46	65.38 ± 3.30	91.49 ± 3.31	36.95 ± 10.7	82.31	54.89
	KLDA	77.46 ± 3.08	57.56 ± 3.46	81.57 ± 2.00	58.08 ± 3.94	77.75 ± 1.69	67.70 ± 2.68	92.87 ± 3.92	31.98 ± 14.1	82.41	53.83
	FECAM	78.14 ± 4.38	59.20 ± 4.72	84.59 ± 0.54	53.98 ± 1.36	80.12 ± 1.50	65.35 ± 2.19	**95.48** ± 2.25	**24.45** ± 7.70	84.58	50.74
	Residuals	73.21 ± 4.69	66.04 ± 3.92	79.35 ± 1.22	62.69 ± 1.55	78.74 ± 2.07	66.44 ± 1.67	91.24 ± 4.51	38.06 ± 13.3	80.64	58.31
	NECO	73.75 ± 3.89	65.48 ± 3.56	79.65 ± 0.87	62.73 ± 1.25	79.87 ± 1.31	64.99 ± 1.82	90.32 ± 4.19	42.42 ± 10.8	80.90	58.91
	CODA-P	59.19 ± 0.88	85.60 ± 0.55	64.71 ± 0.04	82.11 ± 0.09	64.69 ± 0.07	82.36 ± 1.79	75.96 ± 0.65	71.73 ± 0.86	66.14	80.45
	SLCA	78.34 ± 0.21	75.12 ± 0.32	77.95 ± 0.11	73.65 ± 1.26	74.79 ± 1.50	76.83 ± 2.38	86.76 ± 0.95	55.48 ± 3.38	79.46	70.27
	RanPac	85.17 ± 1.05	62.52 ± 1.46	86.15 ± 0.03	56.47 ± 0.25	84.10 ± 0.11	64.65 ± 0.65	94.30 ± 0.43	35.62 ± 3.39	87.43	54.82
	AnaNC (ours)	**85.54** ± 2.49	**52.86** ± 1.90	**87.12** ± 0.13	**51.71** ± 0.96	**85.67** ± 0.53	**59.32** ± 1.36	94.81 ± 0.84	31.31 ± 5.16	**88.29**	**48.80**

Table 3: OOD detection performance under the cross-dataset setting using CIFAR-100 as the ID dataset. CIFAR-10 and Tiny-ImageNet are used as near-OOD datasets, and Places365 and FashionMNIST as far-OOD datasets. Results are averaged over three random class-task splits, with standard deviation reported.

Method	CIFAR100		ImageNet-R		CUB		Cars		Average	
	A_{avg} ↑	A_{last} ↑	A_{avg} ↑	A_{last} ↑	A_{avg} ↑	A_{last} ↑	A_{avg} ↑	A_{last} ↑	A_{avg} ↑	A_{last} ↑
NCM	86.55 ± 0.03	79.07 ± 0.87	70.96 ± 0.12	63.43 ± 0.57	77.34 ± 1.60	69.51 ± 0.32	53.44 ± 1.82	41.43 ± 1.70	72.07	63.36
MD	89.60 ± 0.03	83.10 ± 0.34	75.30 ± 0.24	68.20 ± 0.14	83.72 ± 0.42	79.84 ± 0.14	81.39 ± 0.61	**78.15** ± 0.20	82.50	77.32
KLDA	90.76 ± 0.14	84.75 ± 0.29	72.58 ± 0.13	67.09 ± 0.39	84.60 ± 0.56	78.11 ± 0.55	80.66 ± 0.47	72.16 ± 0.14	82.15	75.52
FECAM	90.28 ± 0.09	84.65 ± 0.35	65.28 ± 0.58	58.06 ± 0.48	81.62 ± 0.73	74.55 ± 0.67	72.63 ± 1.00	63.49 ± 0.67	77.45	70.19
CODA-P	84.57 ± 1.24	76.85 ± 0.22	72.57 ± 1.37	65.71 ± 0.12	65.83 ± 1.40	54.01 ± 1.14	44.44 ± 1.86	32.90 ± 1.66	66.85	57.37
SLCA	88.10 ± 0.19	82.08 ± 0.24	72.54 ± 1.72	65.90 ± 1.33	85.60 ± 0.25	79.73 ± 0.29	79.13 ± 1.19	72.80 ± 0.87	81.34	75.13
RanPac	91.48 ± 0.21	86.38 ± 0.19	75.63 ± 0.01	71.97 ± 0.63	80.47 ± 0.90	70.83 ± 0.63	70.70 ± 0.62	63.78 ± 0.22	79.57	73.24
AnaNC (ours)	**91.67** ± 0.11	**86.41** ± 0.26	**77.82** ± 0.16	**72.44** ± 0.39	**86.87** ± 0.56	**81.49** ± 0.13	**83.43** ± 0.38	76.53 ± 0.15	**84.95**	**79.22**

Table 4: CIL performance with the DINO backbone across different datasets. Results are averaged over three random class-task splits, with standard deviation reported. Note that Residuals and NECO are not included as they are designed only for OOD detection and cannot perform CIL.

ter count is approximately 33.25M, which remains modest compared to the size of the PTM.

AnaNC is highly efficient as well. For example, on CIFAR-100 with our setup, FSA requires about 6 minutes, feature extraction for all training samples takes around 3 minutes, while all AnaNC operations during training take less than 3 seconds, making PTM the dominant computational bottleneck.

Class-incremental Learning Results

Although our main experiments focused on continual OOD detection, CIL is also of significant importance. The same NCM strategy, after creating the NC properties, can be applied to CIL. The results are reported in Table 4 using the DINO backbone. Again, AnaNC markedly outperforms the baselines, achieving an average improvement of 2.45% in A_{avg} and 1.90% in A_{last}. These results demonstrate that AnaNC can effectively address the dual challenges of CIL and continual OOD detection.

Conclusion

This paper focused on continual OOD detection and learning. While prototype-based methods offer a promising approach, this paper proposed a stronger approach by leveraging the theoretical properties of NC. In CL, catastrophic forgetting makes it difficult to realize these properties through training. To overcome this challenge, we introduced a novel method that analytically imposes NC properties in the feature space of a frozen PTM, eliminating the need for further training and avoiding the catastrophic forgetting it would cause. Extensive experiments demonstrate that leveraging the resulting feature geometry with NCM enables our method to outperform state-of-the-art baselines in both continual OOD detection and CIL.

Limitations: While our work focuses on CIL, it can also be applied to TIL, where the task identity is available. However, we do not address domain-incremental learning (DIL), where the classes remain fixed but data distribution shifts over time. We believe our framework can be extended to DIL with minor adaptations, which we leave for future work.

Acknowledgments

The work of Saleh Momeni and Bing Liu was supported in part by two NSF grants (IIS-2229876 and CNS-2225427), and an NVIDIA's Academia Grant, which provided cloud compute via its Saturn Cloud.

References

Aguilar, E.; Raducanu, B.; Radeva, P.; and Van de Weijer, J. 2023. Continual evidential deep learning for out-of-distribution detection. In *CVPR*.

Aljundi, R.; Caccia, L.; Belilovsky, E.; Caccia, M.; Lin, M.; Charlin, L.; and Tuytelaars, T. 2019. Online continual learning with maximally interfered retrieval. *arXiv preprint arXiv:1908.04742*.

Aljundi, R.; Reino, D. O.; Chumerin, N.; and Turner, R. E. 2022. Continual novelty detection. In *Conference on Lifelong Learning Agents*, 1004–1025. PMLR.

Ammar, M. B.; Belkhir, N.; Popescu, S.; Manzanera, A.; and Franchi, G. 2023. NECO: NEural Collapse Based Out-of-distribution detection. *CoRR*.

Bendale, A.; and Boult, T. 2015. Towards open world recognition. In *CVPR*, 1893–1902.

Buzzega, P.; Boschini, M.; Porrello, A.; Abati, D.; and Calderara, S. 2020. Dark experience for general continual learning: a strong, simple baseline. *NeurIPS*.

Caron, M.; Touvron, H.; Misra, I.; Jégou, H.; Mairal, J.; Bojanowski, P.; and Joulin, A. 2021. Emerging properties in self-supervised vision transformers. In *CVPR*.

Chaudhry, A.; Rohrbach, M.; Elhoseiny, M.; Ajanthan, T.; Dokania, P.; Torr, P.; and Ranzato, M. 2019. On Tiny Episodic Memories in Continual Learning. *arXiv:1902.10486*.

Chen, X.; Xie, S.; and He, K. 2021. An Empirical Study of Training Self-Supervised Vision Transformers. In *ICCV*.

Cook, M.; Zare, A.; and Gader, P. 2020. Outlier detection through null space analysis of neural networks. *arXiv preprint arXiv:2007.01263*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*.

Fei, G.; and Liu, B. 2016. Breaking the closed world assumption in text classification. In *NAACL-HLT*.

Ghassemi, N.; and Fazl-Ersi, E. 2022. A Comprehensive Review of Trends, Applications and Challenges In Out-of-Distribution Detection. *arXiv preprint arXiv:2209.12935*.

Goswami, D.; Liu, Y.; Twardowski, B.; and Van De Weijer, J. 2023. Fecam: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. *NeurIPS*.

Gower, J. C.; and Dijksterhuis, G. B. 2004. *Procrustes problems*, volume 30. Oxford university press.

Gummadi, M.; Kent, D.; Mendez, J. A.; and Eaton, E. 2022. Shels: Exclusive feature sets for novelty detection and continual learning without class boundaries. In *CoLLaS*.

Haas, J.; Yolland, W.; and Rabus, B. T. 2023. Linking Neural Collapse and L2 Normalization with Improved Out-of-Distribution Detection in Deep Neural Networks. *Transactions on Machine Learning Research*.

Hayes, T. L.; and Kanan, C. 2020. Lifelong machine learning with deep streaming linear discriminant analysis. In *CVPR workshops*.

He, J.; and Zhu, F. 2022. Out-of-distribution detection in unsupervised continual learning. In *CVPR*.

Hendrycks, D.; Basart, S.; Mazeika, M.; Zou, A.; Kwon, J.; Mostajabi, M.; Steinhardt, J.; and Song, D. 2022. Scaling Out-of-Distribution Detection for Real-World Settings. In *ICML*.

Hendrycks, D.; Basart, S.; Mu, N.; Kadavath, S.; Wang, F.; Dorundo, E.; Desai, R.; Zhu, T.; Parajuli, S.; Guo, M.; et al. 2021. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*.

Hendrycks, D.; and Gimpel, K. 2017. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In *ICLR*.

Huang, G.-B.; Zhou, H.; Ding, X.; and Zhang, R. 2011. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2): 513–529.

Kaymak, D.; Kim, G.; Kaichi, T.; Konishi, T.; and Liu, B. 2025. Learning After Model Deployment. In *ECAI*.

Ke, Z.; Liu, B.; Ma, N.; Xu, H.; and Shu, L. 2021. Achieving Forgetting Prevention and Knowledge Transfer in Continual Learning. *NeurIPS*.

Kim, G.; Xiao, C.; Konishi, T.; Ke, Z.; and Liu, B. 2022. A theoretical study on solving continual learning. *NeurIPS*.

Kim, G.; Xiao, C.; Konishi, T.; Ke, Z.; and Liu, B. 2025. Open-world continual learning: Unifying novelty detection and continual learning. *Artificial Intelligence*, 338: 104237.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13).

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

Lee, K.; Lee, K.; Lee, H.; and Shin, J. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *NeurIPS*.

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

Liang, N.-Y.; Huang, G.-B.; Saratchandran, P.; and Sundararajan, N. 2006. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on neural networks*, 17(6).

Liang, S.; Li, Y.; and Srikant, R. 2018. Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks. In *ICLR*.

Liang, Y.-S.; and Li, W.-J. 2024. Inflora: Interference-free low-rank adaptation for continual learning. In *CVPR*.

Liu, W.; Wang, X.; Owens, J.; and Li, Y. 2020. Energy-based out-of-distribution detection. *NeurIPS*.

Liu, Y.; Zhao, W.; and Guo, Y. 2025. H2ST: Hierarchical Two-Sample Tests for Continual Out-of-Distribution Detection. In *CVPR*.

Malinin, A.; and Gales, M. 2019. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. *NeurIPS*.

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

McDonnell, M. D.; Gong, D.; Parvaneh, A.; Abbasnejad, E.; and van den Hengel, A. 2023. Ranpac: Random projections and pre-trained models for continual learning. *NeurIPS*.

Miao, W.; Pang, G.; Nguyen, T.-T.; Fang, R.; Zheng, J.; and Bai, X. 2025. OpenCIL: Benchmarking Out-of-Distribution Detection in Class Incremental Learning. *Pattern Recognition*, 112163.

Momeni, S.; Mazumder, S.; and Liu, B. 2025. Continual learning using a kernel-based method over foundation models. In *AAAI*.

Papyan, V.; Han, X.; and Donoho, D. L. 2020. Prevalence of neural collapse during the terminal phase of deep learning training. *PNAS*, 117(40): 24652–24663.

Peng, L.; Elenter, J.; Agterberg, J.; Ribeiro, A.; and Vidal, R. 2025. LoRanPAC: Low-rank Random Features and Pre-trained Models for Bridging Theory and Practice in Continual Learning. In *ICLR*.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. In *CVPR*.

Ren, J.; Fort, S.; Liu, J.; Roy, A. G.; Padhy, S.; and Lakshminarayanan, B. 2021. A simple fix to mahalanobis distance for improving near-ood detection. *arXiv:2106.09022*.

Rios, A.; Ahuja, N.; Ndiour, I.; Genc, U.; Itti, L.; and Tickoo, O. 2022. incDFM: Incremental Deep Feature Modeling for Continual Novelty Detection. In *ECCV 2022*.

Roy, A.; Moulick, R.; Verma, V. K.; Ghosh, S.; and Das, A. 2024. Convolutional prompting meets language models for continual learning. In *CVPR*.

Serra, J.; Suris, D.; Miron, M.; and Karatzoglou, A. 2018. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*.

Smith, J. S.; Karlinsky, L.; Gutta, V.; Cascante-Bonilla, P.; Kim, D.; Arbelle, A.; Panda, R.; Feris, R.; and Kira, Z. 2023. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *CVPR*.

Sun, H.-L.; Zhou, D.-W.; Zhao, H.; Gan, L.; Zhan, D.-C.; and Ye, H.-J. 2025. Mos: Model surgery for pre-trained model-based class-incremental learning. In *AAAI*.

Sun, Y.; Guo, C.; and Li, Y. 2021. React: Out-of-distribution detection with rectified activations. *NeurIPS*.

Sun, Y.; and Li, Y. 2022. Dice: Leveraging sparsification for out-of-distribution detection. In *ECCV*.

Sun, Y.; Ming, Y.; Zhu, X.; and Li, Y. 2022. Out-of-distribution detection with deep nearest neighbors. In *ICML*.

Van de Ven, G. M.; and Tolias, A. S. 2019. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset.

Wang, F.-Y.; Zhou, D.-W.; Ye, H.-J.; and Zhan, D.-C. 2022a. Foster: Feature boosting and compression for class-incremental learning. In *ECCV*.

Wang, H.; Li, Z.; Feng, L.; and Zhang, W. 2022b. ViM: Out-Of-Distribution with Virtual-logit Matching. In *CVPR*.

Wang, L.; Xie, J.; Zhang, X.; Huang, M.; Su, H.; and Zhu, J. 2023. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. *NeurIPS*.

Wang, L.; Zhang, X.; Su, H.; and Zhu, J. 2024. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8).

Wang, Y.; Li, B.; Che, T.; Zhou, K.; Liu, Z.; and Li, D. 2021. Energy-based open-world uncertainty modeling for confidence calibration. In *ICCV*.

Wang, Z.; Zhang, Z.; Lee, C.-Y.; Zhang, H.; Sun, R.; Ren, X.; Su, G.; Perot, V.; Dy, J.; and Pfister, T. 2022c. Learning to prompt for continual learning. In *CVPR*.

Wortsman, M.; Ramanujan, V.; Liu, R.; Kembhavi, A.; Rastegari, M.; Yosinski, J.; and Farhadi, A. 2020. Supermasks in superposition. *NeurIPS*.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Yang, J.; Wang, P.; Zou, D.; Zhou, Z.; Ding, K.; Peng, W.; Wang, H.; Chen, G.; Li, B.; Sun, Y.; et al. 2022. Openood: Benchmarking generalized out-of-distribution detection. *NeurIPS*.

Yang, L.; Luo, P.; Change Loy, C.; and Tang, X. 2015. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*.

Yang, Y.; Yuan, H.; Li, X.; Lin, Z.; Torr, P.; and Tao, D. 2023. Neural Collapse Inspired Feature-Classifier Alignment for Few-Shot Class-Incremental Learning. In *ICLR*.

Zeng, G.; Chen, Y.; Cui, B.; and Yu, S. 2019. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8): 364–372.

Zhang, G.; Wang, L.; Kang, G.; Chen, L.; and Wei, Y. 2023. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *ICCV*.

Zhou, B.; Lapedriza, A.; Khosla, A.; Oliva, A.; and Torralba, A. 2017. Places: A 10 million image database for scene recognition. *IEEE TPAMI*, 40(6): 1452–1464.

Zhou, D.-W.; Sun, H.-L.; Ning, J.; Ye, H.-J.; and Zhan, D.-C. 2024a. Continual learning with pre-trained models: a survey. In *IJCAI*.

Zhou, D.-W.; Ye, H.-J.; Zhan, D.-C.; and Liu, Z. 2024b. Revisiting Class-Incremental Learning with Pre-Trained Models: Generalizability and Adaptivity are All You Need. *International Journal of Computer Vision*.

Zhuang, H.; Weng, Z.; Wei, H.; Xie, R.; Toh, K.-A.; and Lin, Z. 2022. ACIL: Analytic class-incremental learning with absolute memorization and privacy protection. *NeurIPS*.