KAC: Kolmogorov-Arnold Classifier for Continual Learning

Yusong Hu[1], Zichen Liang[1], Fei Yang[1,2], Qibin Hou[1,2], Xialei Liu[1,2*], Ming-Ming Cheng[1,2]
[1]VCIP, CS, Nankai University [2]NKIARI, Shenzhen Futian
{ethanhu, liangzc}@mail.nankai.edu.cn
{feiyang, houqb, xialei, cmm}@nankai.edu.cn

Abstract

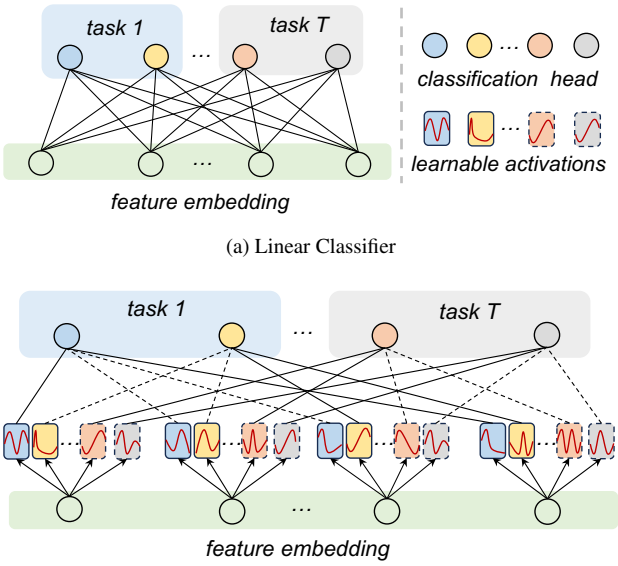
Continual learning requires models to train continuously across consecutive tasks without forgetting. Most existing methods utilize linear classifiers, which struggle to maintain a stable classification space while learning new tasks. Inspired by the success of Kolmogorov-Arnold Networks (KAN) in preserving learning stability during simple continual regression tasks, we set out to explore their potential in more complex continual learning scenarios. In this paper, we introduce the Kolmogorov-Arnold Classifier (KAC), a novel classifier developed for continual learning based on the KAN structure. We delve into the impact of KAN's spline functions and introduce Radial Basis Functions (RBF) for improved compatibility with continual learning. We replace linear classifiers with KAC in several recent approaches and conduct experiments across various continual learning benchmarks, all of which demonstrate performance improvements, highlighting the effectiveness and robustness of KAC in continual learning. The code is available at https://github.com/Ethanhuhuhu/KAC.

1. Introduction

Deep learning models are typically trained on a fixed dataset in a single session, achieving impressive performance on various static tasks. In contrast, real-world scenarios continuously evolve, necessitating models that can learn incrementally from a data stream. However, in such scenarios, these models often encounter a significant challenge, known as catastrophic forgetting [12]. Continual learning [2, 8, 18, 47] investigates how to effectively train models in such dynamic environments with sequential data exposure, aiming to adapt and avoid forgetting over time.

Class incremental learning (CIL) [50], as a key challenge in continual learning, has garnered extensive research interest. It involves the continuous introduction of new classes with ongoing tasks, requiring the model to conduct classification on all encountered classes after training on new tasks. Most CIL methods retain exemplars and employ techniques, such as knowledge distillation [9, 50, 59] or dynamic architectures [6, 10, 31, 60], to mitigate forgetting. With the development of pre-trained models, numerous studies [44, 63] have attempted to explore the applications of pre-trained models in CIL, achieving impressive results. Among these, prompt-based approaches [16, 52, 57, 58] have attracted considerable attention.

For existing methods, some [19, 44, 62] focus on fea-



(a) Linear Classifier

(b) Kolmogorov-Arnold Classifier

Figure 1. Brief comparison between conventional linear classifier and our Kolmogorov-Arnold classifier. The solid lines represent activated weights, while the dashed ones represent suppressed weights. (a) Conventional linear classifiers activate each weight equally across all tasks, resulting in irrelevant weights being equally updated in the new task.. (b) our Kolmogorov-Arnold Classifier learns class-specific learnable activations for each channel across all categories, minimizing forgetting caused by irrelevant weight changes.

*Corresponding author.

ture space design through carefully crafted classifiers and training or inference strategies, achieving excellent performance. These studies demonstrate that a well-structured feature space can effectively mitigate the forgetting issue in that a stable distribution is crucial for continual classification tasks while the design of classifiers is essential for constructing the feature space and reducing forgetting in continuous tasks. However, most existing approaches [16, 52, 64] use linear classifiers or nearest class mean classifiers (NCM) [50], with limited research focused on developing a specific classifier for CIL to effectively mitigate catastrophic forgetting, which warrants further study.

Recently, a novel architecture, Kolmogorov–Arnold Networks (KAN) [41] , has been proposed, demonstrating natural effectiveness in continual learning. The authors compare KAN with Multi-Layer Perceptrons (MLP) [25] on a toy continual 1D regression problem, which requires the model to fit 5 Gaussian peaks sequentially. KAN exhibits superior performance, effectively mitigating catastrophic forgetting, attributed to the locality of splines and inherent local plasticity. This locality allows KAN to identify relevant regions for re-organization while maintaining stability in other areas during sequential tasks [41]. These findings motivate us to explore the applications of KAN in more challenging CIL tasks.

In this paper, we introduce the Kolmogorov-Arnold Classifier (KAC), a plug-and-play classifier for Continual Learning built upon the KAN architecture. Leveraging the Kolmogorov-Arnold representation theorem [34], we integrate learnable activation functions on the edges of the classifier. We find that the conventional KAN with B-spline functions struggles with high-dimensional data, resulting in insufficient model plasticity, which may reduce the model's adaptability when directly employed as a classifier. This limitation forces models to undergo excessive updates when learning new tasks, resulting in significant forgetting.

To address this, we explore spline functions and identify Radial Basis Functions (RBF) as an effective alternative for continual learning. By utilizing RBF in our KAC, we enhance the model's ability to adapt CIL while minimizing forgetting. Thanks to these learnable spline activations, the KAC allows the model to select specific activation ranges of interest for each channel while preserving the distribution of other parts, and RBF makes it more compatible with CIL. As shown in Fig. 1b, these learnable activations help the model select interesting parts of each channel and activate them for determination rather than activating all edges like a simple linear classifier in Fig. 1a. This brings notable benefits to class incremental learning. When new tasks arrive, the learnable activation functions assist the model in selecting relevant parts of each channel for updating. This prevents the drift of irrelevant features during the training process for the new tasks. Meanwhile, the deactivated portions of the old tasks remain unaffected by these updates, reducing the forgetting of old tasks.

To demonstrate the superiority of KAC, we conduct experiments on several prompt-based continual learning approaches, which are built upon a pre-trained backbone where the classifiers play a key role in these approaches. The models employing our method achieve significant improvement across various CIL scenarios on multiple datasets by simply replacing the linear classifier with our KAC without making any other modifications or hyperparameter adjustments. Additionally, experiments conducted in the Domain Incremental Learning (DIL) [56] setting reveal that our method can also improve performance, demonstrating its effectiveness and robustness.

Our main contributions can be summarised as follows:

- We explore the application of Kolmogorov-Arnold Networks (KAN) in continual learning and analyze its weaknesses when employed in continual learning and how to enhance its compatibility with such tasks.
- We introduce the Kolmogorov-Arnold Classifier (KAC), a novel continual classifier based on the KAN structure with Radial Basis Functions (RBF) as its basis functions. KAC enhances the stability and plasticity of CIL approaches.
- We integrate our KAC into various approaches and validate their performance across multiple continual learning benchmarks. The results demonstrate that KAC can effectively reduce forgetting in these methods.

2. Related Work

Class Incremental Learning aims to learn a sequence of classification tasks sequentially, where the number of classes increases with each task. The primary challenge in it is catastrophic forgetting[43]. Several studies work on it and they can be broadly categorized into three main strategies: regularization-based, structure-based, and replay-based methods. Regularization-based methods reduce forgetting by employing knowledge distillation techniques[9, 59, 61] or imposing constraints on key model parameters[29, 32, 40]. Structure-based methods[6, 10, 27, 54] mitigate forgetting through dynamic network architectures. Replay-based methods retain a small portion of old data[28, 50] or use auxiliary models[14, 30, 51] to generate synthetic data, which are combined with new-class data to update the model.

CIL with Pre-trained Models have demonstrated their competitive performance in Class Incremental Learning due to their strong transferability. Techniques such as LAE [13] and SLCA [63] enhance model adaptation through EMA-based updates and dynamic classifier adjustments. RanPAC [44] employs random projection to improve continual learning, while EASE [64] focuses on optimizing task-

specific, expandable adapters to enhance knowledge retention. Benefiting from parameter-efficient tuning in NLP, prompt-based methods have achieved promising results in Class Incremental Learning. These approaches utilize adaptive prompts to guide frozen transformer models, facilitating efficient task-specific learning without modifying encoder parameters. Techniques like L2P [58], DualPrompt [57], S-Prompts [56], CODA-Prompt [52], HiDe-Prompt [55], and CPrompt [15] introduce diverse prompt selection strategies to improve task learning, knowledge retention, and model robustness.

Kolmogorov-Arnold Networks (KAN) [37] is a novel network architecture based on the Kolmogorov-Arnold representation theorem [34]. It represents multivariate functions as combinations of multiple univariate functions and uses nonlinear spline functions for approximation. Some explorations focus on how to apply KAN to solve scientific problems [3, 26, 33], while others seek various basis functions to enhance performance [1, 3, 37]. Many works [4, 7, 17, 42] apply KAN across various fields and investigate how to effectively leverage its advantages in these domains. These studies encourage us to explore the application of KAN in continual learning.

3. Method

3.1. Preliminaries

Class Incremental Learning. In Class Incremental Learning (CIL), a model needs to learn classes step by step. At each step t, the model needs to learn the classes specific to that step, denoted as $\mathcal{Y}_t$, with only access to the current dataset $D_t = \{(\boldsymbol{x}_t^i, y_t^i)\}_{i=1}^{n_t}$, where $\boldsymbol{x}_t^i$ represents an input image and y_t^i is its corresponding label. A key challenge in CIL is how to maintain the stability of the model to avoid catastrophic forgetting [12] while learning new tasks. With a model consisting of a backbone F, and a classifier $h \in \mathbb{R}^{n \times C}$, where n denotes the embedding dimension and C represents the total number of learned classes, the model is tasked with predicting the class label $y = h(F(\boldsymbol{x})) \in \mathcal{Y}$ for test samples from new classes as well as samples from previously encountered tasks.

Kolmogorov–Arnold Networks. Kolmogorov–Arnold Network (KAN) [41] is a novel model architecture that serves as a promising alternative to multi-layer perceptrons (MLPs) [21, 25]. While MLPs rely on the Universal Approximation Theorem (UAT) [25], KANs are inspired by the Kolmogorov-Arnold representation Theorem (KAT) [34]. KAT posits that any multivariate continuous function $f(\boldsymbol{x})$ defined on a bounded domain can be expressed as a finite composition of univariate continuous functions through addition. The Kolmogorov-Arnold representation theorem can be written as:

$$f(\boldsymbol{x}) = f(x_1, x_2, ..., x_n) = \sum_{q=1}^{2n+1} \Phi_q \Big(\sum_{p=1}^{n} \phi_{q,p}(x_p) \Big), \quad (1)$$

in which Φ_q and $\phi_{q,p}$ are univariate functions for each variable. KAN parametrizes the $\phi_{q,p}$ and Φ_q as B-spline curves, with learnable coefficients of local B-spline basis functions $B(\boldsymbol{x})$ [49]. In practice, a residual connection, consisting of a linear function with activation $b(\boldsymbol{x}) = silu(\boldsymbol{x}) = \boldsymbol{x}/(1 + e^{-\boldsymbol{x}})$, is linearly combined with the B-spline curve $spline(\boldsymbol{x}) = \sum_i \omega_i B_i(\boldsymbol{x})$ to form the final ϕ:

$$\phi(x) = \omega_b b(\boldsymbol{x}) + \omega_s spline(\boldsymbol{x}), \quad (2)$$

where the ω_b and ω_s represent the linear functions that control the overall magnitude of the activation function. Consequently, a KAN layer can be expressed as:

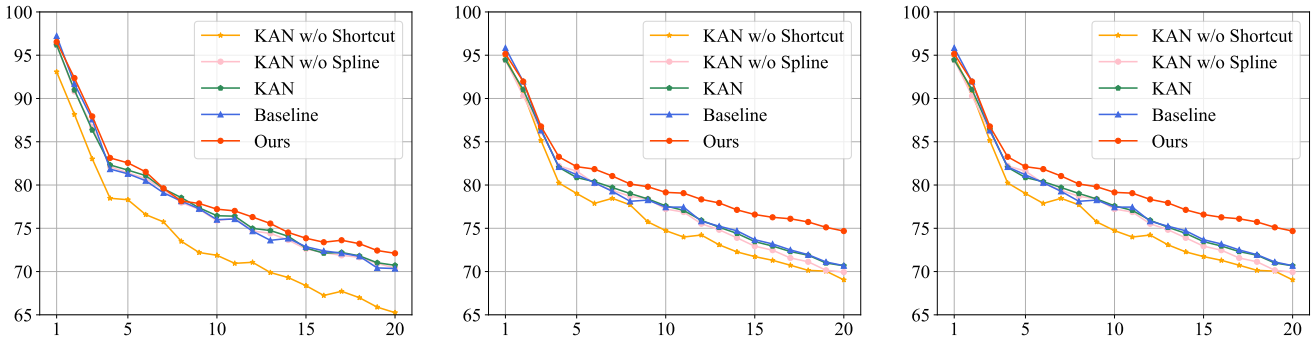
$$\boldsymbol{x}_{l+1} = \underbrace{\begin{pmatrix} \phi_{l,1,1}(.) & \phi_{l,1,2}(.) & \cdots & \phi_{l,1,n_l}(.) \\ \phi_{l,2,1}(.) & \phi_{l,2,2}(.) & \cdots & \phi_{l,2,n_l}(.) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{l,n_{l+1},1}(.) & \phi_{l,n_{l+1},2}(.) & \cdots & \phi_{l,n_{l+1},n_l}(.) \end{pmatrix}}_{\Phi_l} \boldsymbol{x}_l. \quad (3)$$

The $\boldsymbol{x}_l$ and $\boldsymbol{x}_{l+1}$ represent the input and output of a KAN layer, while ϕ_l is the 1D univariate function matrix for each layer. The KAN networks are constructed by stacking multiple KAN layers.

3.2. Conventional KAN layer is not a good continual classifier

In [41], the authors present experimental results from a toy 1D regression task, demonstrating that the locality of splines can inherently avoid catastrophic forgetting. This insight inspires us to introduce KAN to CIL. A straightforward way to leverage the locality of KAN is directly utilizing a KAN layer to develop a continual classifier, replacing the linear classifier in CIL methods. To achieve this, we simply replace the linear classifier $h(\boldsymbol{x})$ with a KAN layer that has an input dimension of d and an output dimension of C. We compared their performances across several baseline methods. The experimental results are shown in Fig. 2, demonstrating that the simple substitution of replacing the linear classifier with a KAN layer does not lead to any improvement, even achieving worse performance.

We decompose the KAN layer into two parts: the residual connection $b(\boldsymbol{x})$ and the B-spline curve $spline(\boldsymbol{x})$ and individually replace the linear classifier with these two components to investigate why directly introducing the KAN layer increases forgetting. A surprising finding is that the B-spline functions lead to a severe performance drop across all baselines.



(a) L2P (b) DualPrompt (c) CODAPrompt

Figure 2. Comparison of the accuracy curves of three recent approaches with different classifiers in the ImageNet-R 20-step scenario. The x-axis represents the increasing number of tasks, while the y-axis shows the corresponding test accuracy at each step. The Baseline indicates performance with a conventional linear classifier, while the other curves represent results with ablated KAN classifiers and our Kolmogorov-Arnold Classifier.

To understand why the B-spline curve replacing the conventional linear classifier leads to severe forgetting, we need to delve deeper into the differences between linear layers and splines. In high-dimensional complex data, spline functions encounter the curse of dimensionality (COD) [20]; as the data dimensionality increases, the model struggles with data approximation [22, 35, 45]. This is because splines cannot effectively model the compositional structure present in the data, while linear classifiers benefit from their fully connected structure, allowing them to learn this structure effectively [23]. Although KAN networks mitigate COD through approximation theory [41] by stacking KAN layers, approximating high-dimensional function remains a challenging problem for a single spline layer, whereas it is relatively straightforward for conventional linear classifiers.

It is precisely the weak fitting ability of B-spline functions on high-dimensional data that leads to severe forgetting when it is introduced into CIL. In CIL, a network typically consists of a backbone F that encodes images to feature embeddings and a classification head h, which serves as a high-dimensional projection mapping the embeddings to class probabilities. Most methods accommodate new classes by adding classifiers while sharing the backbone across all tasks. The final logits l for classification are always calculated as:

$$l = h(F(\boldsymbol{x})), h = [h_1, h_2, \cdots, h_t]. \tag{4}$$

To prevent significant forgetting caused by changes in the backbone that affect the feature space, the model must maintain stable backbone parameters during training on new tasks. Consequently, many methods use regularization techniques to restrict changes in feature embeddings [31, 38, 59, 61]. However, due to the limited approximation capability of a single B-spline layer, the model requires more extensive updates to the backbone parameters compared to conventional linear classifiers to achieve good performance on new tasks. This extensive updating can severely disrupt the feature space, leading to pronounced forgetting.

Based on the above analysis, we believe that the weak fitting ability of a single B-spline function prevents the model from leveraging the locality of the KAN layer. Therefore, we need to enhance the spline function's fitting ability to adapt the KAN structure to CIL tasks. [36, 39] indicates that, in specific senses, a shallow KAT-based layer can break the COD problem when approximating high-dimensional functions through designed basis functions with particular compositional structures, motivates us to explore the types of basis functions that are compatible with CIL.

3.3. Radial basis function is great for class incremental learning

Several studies [44, 62, 65] assume that the classification space follows a Gaussian space and develop approaches based on this premise, achieving excellent performance. It suggests that building a Gaussian classification space can help models effectively learn new tasks while combating catastrophic forgetting. Can we find a kind of basis function in this sense that allows a KAT-based layer function as a continual classifier, addressing the COD problem and benefiting CIL? The answer is yes!

FastKAN [37] proves that the B-splines basis function in KAN [41] can be well replaced by Radial Basis Functions (RBF) [5, 46]. We find this substitution brings more benefits to CIL when KAN is introduced as a continual classifier as shown later. A KAN layer with RBF is represented as:

$$f(\boldsymbol{x}) = \sum_{p=1}^{n} \Phi_p \sum_{i=1}^{N} \omega_{p,i} \phi(||x_p - c_i||), \tag{5}$$

where c_i represents a series of center points evenly dis-

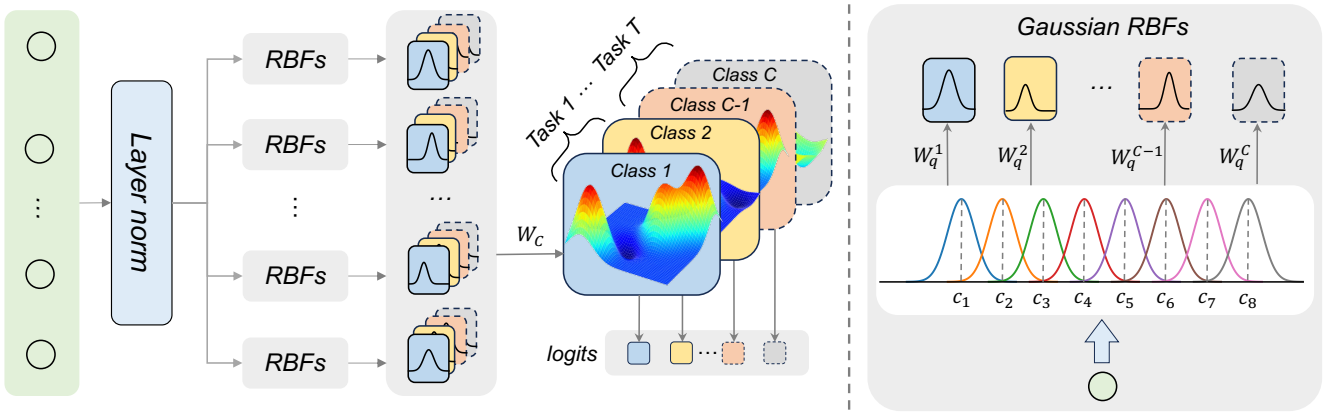


Figure 3. An overview of the pipeline of the proposed Kolmogorov-Arnold Classifier. For the input feature embeddings, we first normalize them using a layer normalization, then pass them through a set of RBFs that activate them to learnable Gaussian distributions. Finally, we weight all channels with W_C to obtain the decision space for each class. The right side shows the process of Gaussian RBFs, which map univariate variables to different Gaussian distributions centered at various points and weight these distributions with W_q^c to derive the final activation distribution for each channel across all classes. The output logits are sampled based on the channel values within the distribution of each class. As tasks increase, new classes can be accommodated by simply expanding W_C.

tributed within a specific range, with N denoting the total number of c_i. And $\phi(.)$ is an RBF served as the basis functions whose value solely depends on the distance between input x_p and center point c_i. The term $\omega_{p,i}$ denotes the weight for each ϕ. A Gaussian function with covariance σ_i can be chosen as ϕ while it's defined as:

$$\phi(||x_p - c_i||) = \exp\Big(-\frac{(x_p - c_i)^2}{2\sigma_i^2}\Big). \tag{6}$$

While introducing the Gaussian RBF function as the basis function of KAN demonstrates faster evaluation speeds and enhanced performance, as shown in [37], an inherent Gaussian structure is also established with it, which can serve as an effective compositional structure for CIL scenarios. With a series of Gaussian distributions $\mathcal{N}$ centered at $c = [c_1, c_2, \cdots, c_N]$, the activation function for each dimension is formed by combining N Gaussian distributions, and the distribution of each dimension can be represented as a Gaussian mixture model:

$$\begin{aligned}\sum_{i=1}^{N} \omega_{p,i}\phi(\|x_p - c_i\|) &\sim \sum_{i=1}^{N} \omega_{p,i}\mathcal{N}(c_i, \sigma_i^2) \\ &= \omega_{p,1}\mathcal{N}(c_1, \sigma_1^2) + \omega_{p,2}\mathcal{N}(c_2, \sigma_2^2) \\ &\quad + \cdots + \omega_{p,N}\mathcal{N}(c_N, \sigma_N^2)\end{aligned} \tag{7}$$

This mixture formulation preserves the multi-modal characteristics of the original Gaussian components. The final prediction for each class is then expressed as a dimension-wise weighted combination of these Gaussian mixtures:

$$f(\boldsymbol{x}) = \sum_{p=1}^{n} \Phi_p \left[\sum_{i=1}^{N} \omega_{p,i} \exp\left(-\frac{(x_p - c_i)^2}{2\sigma_i^2}\right)\right] \tag{8}$$

We can easily derive that, thanks to the introduction of Gaussian RBF functions, the features of pth dimension in the KAN layer, after the activation function, follow a Gaussian mixture distribution. When we simply define Φ_p as a learnable weight for each dimension, it is evident that the resulting function form conforms to the Gaussian Process (GP) with first-order additive kernels defined in [11]. This structure is consistently easy to fit for classification tasks and possesses a strong long-range structure to effectively address the COD problem when approximating high-dimensional functions [11]. With functions like this serving as the basis functions for continual classifiers, it not only projects each channel of the feature into a Gaussian space but also allows the model to select an interested range for each channel tailored to different classes.

3.4. Kolmogorov-Arnold Classifier for CIL

The above analysis demonstrates that the KAN layer with RBF can benefit CIL, motivating us to introduce our Kolmogorov-Arnold Classifier (KAC), which can be integrated into any CIL approach by replacing the conventional linear classifier with it.

An overview of the KAC is shown in Fig. 3. The KAC firstly regularizes the feature distribution with a Layer Normalization $\mathcal{LN}$, resulting in a normalized embedding $\mathcal{LN}(F(\boldsymbol{x})) = [x_1', x_2', \cdots, x_n']$. After that, it incorporates a KAN layer that includes N Gaussian Radial Basis Func-

Table 1. Results on ImageNet-R dataset. We report the average incremental accuracy and the last accuracy on CIL scenarios of 5, 10, 20, and 40 steps and make comparisons on various approaches, evaluating the results with a linear classifier (baseline) and with our KAC. It demonstrates that our KAC consistently improves their performance, especially in long-sequence tasks. The change is indicated next to the accuracy, with blue representing a decrease and red representing an improvement.

Method	5 steps		10 steps		20 steps		40 steps	
	Avg	Last	Avg	Last	Avg	Last	Avg	Last
L2P	78.42	73.57	79.58	73.10	77.93	70.35	74.28	66.02
w/ KAC	77.98 (-0.44)	73.56 (-0.01)	79.22 (-0.36)	73.14 (+0.04)	78.94 (+1.01)	72.11 (+1.76)	76.34 (+2.06)	69.74 (+3.72)
DualPrompt	79.75	74.57	79.50	72.48	78.35	70.68	74.51	66.31
w/ KAC	79.96 (+0.21)	76.37 (+1.80)	80.72 (+1.22)	75.67 (+3.19)	80.40 (+2.05)	74.68 (+4.00)	76.87 (+2.36)	71.24 (+4.93)
CODAPrompt	82.27	77.62	82.49	77.01	80.92	74.40	76.80	69.34
w/ KAC	83.75 (+1.48)	80.14 (+2.52)	84.43 (+1.94)	79.24 (+2.23)	83.59 (+2.67)	77.94 (+3.54)	79.79 (+2.99)	74.31 (+4.97)
CPrompt	84.07	78.68	83.13	76.80	81.83	74.32	78.98	70.07
w/ KAC	84.51 (+0.44)	79.08 (+0.40)	83.97 (+0.84)	78.07 (+1.27)	82.56 (+0.73)	75.73 (+1.41)	80.89 (+1.91)	72.05 (+1.98)

tions centered at $c = [c_1, c_2, \cdots, c_N]$. With the basis function ϕ is like defined in eq. 6, the logit l is then calculated as:

$$l = \mathrm{KAC}\big(F(\boldsymbol{x})\big) = \mathrm{diag}\Big(W_C \cdot \Phi\Big(\mathcal{LN}\big(F(\boldsymbol{x})\big)\Big) \cdot W_q\Big), \tag{9}$$

where $\mathrm{diag}(.)$ represents extracting the diagonal elements of a matrix and the $\Phi\Big(\mathcal{LN}\big(F(\boldsymbol{x})\big)\Big)$ is the learnable Gaussian RBF and it can be calculated like:

$$\begin{pmatrix} \phi(||x'_1 - c_1||) & \phi(||x'_1 - c_2||) & \cdots & \phi(||x'_1 - c_N||) \\ \phi(||x'_2 - c_1||) & \phi(||x'_2 - c_2||) & \cdots & \phi(||x'_2 - c_N||) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(||x'_n - c_1||) & \phi(||x'_n - c_2||) & \cdots & \phi(||x'_n - c_N||) \end{pmatrix}, \tag{10}$$

in which n is the dimensionality of the input embedding and $W_C \in \mathbb{R}^{C \times n}$ is a learnable weight matrix that serves as an output linear function to predict the probability for each class, corresponding to the Φ_p in conventional KAN, while the $W_q \in \mathbb{R}^{N \times C}$ corresponds to the $\phi_{p,q}$ in conventional KAN to serve as the univariate learnable activation for each channel for every class. In practice, the W_C and W_q can be consolidated into a single weight matrix $W \in \mathbb{R}^{C \times (N \times n)}$, from which the final logit is directly predicted using the basis functions ϕ. The KAC is then represented as:

$$\mathrm{KAC}\big(F(\boldsymbol{x})\big) = W \cdot \mathrm{Reshape}\Big(\Phi\Big(\mathcal{LN}\big(F(\boldsymbol{x})\big)\Big)\Big). \tag{11}$$

The $\mathrm{reshape}(.)$ function flattens the $N \times n$ matrix into a 1D vector to facilitate calculations with W.

In a CIL scenario, T tasks arrive sequentially with class counts $[C_1, C_2, \cdots, C_T]$. KAC expands W to accommodate new classes, similar to conventional classifiers [52]. At the tth step, there is an old classification matrix $W^{t-1} \in \mathbb{R}^{(N \times n) \times C_{old}}$, where $C_{old} = C_1 + C_2 + \cdots + C_{t-1}$, and a new matrix $W^t \in \mathbb{R}^{(N \times n) \times C_t}$, with the final W after the tth step being the concatenation of these two matrices.

4. Experiments

4.1. Benchmarks & Implementations

Benchmarks. We evaluate the CIL scenario and further validate the robustness of our method in Domain Incremental Learning (DIL) [56]. For CIL, we conduct experiments on two commonly used datasets, ImageNet-R [24] and CUB200 [53], each containing 200 classes. Starting with 0 base classes, all classes are separated into 5, 10, 20, and 40 steps to feed the model for training sequentially. For DIL, following Sprompt [56], we split the DomainNet [48] dataset into 6 domains, classifying a total of 345 categories across all tasks. All experiments are conducted in a non-exemplar setting, with no old samples saved for new training. The results of experiments with various seeds are presented in the supplementary materials.

Implementation Details. All experiments are conducted with ViT-B/16 backbones. The numbed of RBFs is set to 4, the centers $[c_1, c_2, \cdots, c_N]$ are evenly distributed between -2 and 2, and the σ in the Gaussian functions is set to 1, allowing for an average division of the range. To validate the effectiveness of KAC, we select four prompt-based CIL approaches L2P [58], DualPrompt [57], CODAPrompt [52] and CPrompt [16] as baselines, all of which have achieved superior performance across various CIL benchmarks. These approaches leverage learnable prompts to extract information from pre-trained backbones and classify the extracted embeddings using linear classifiers. We directly replace their classifiers with KAC with their orig-

Table 2. Results on CUB200 dataset. The average incremental accuracy and the last accuracy are reported. KAC delivers significant improvements for all baselines, especially in long-sequence tasks, highlighting its superior performance on fine-grained datasets.

Method	5 steps		10 steps		20 steps		40 steps	
	Avg	Last	Avg	Last	Avg	Last	Avg	Last
L2P	80.05	76.04	74.02	65.28	63.31	51.78	46.84	35.41
w/ KAC	84.42 (+4.37)	83.80 (+7.76)	81.54 (+7.52)	79.77 (+14.49)	73.70 (+10.39)	70.13 (+18.35)	66.08 (+19.24)	60.43 (+25.02)
DualPrompt	81.84	76.38	75.10	64.60	66.89	54.68	50.61	37.55
w/ KAC	86.20 (+4.36)	85.03 (+8.65)	82.18 (+7.08)	79.61 (+14.01)	76.93 (+10.04)	71.91 (+17.23)	71.31 (+20.70)	64.69 (+27.14)
CODAPrompt	83.09	78.73	79.30	71.87	69.49	58.00	52.57	37.81
w/ KAC	86.56 (+3.47)	85.61 (+6.88)	85.04 (+5.74)	82.59 (+10.72)	77.23 (+7.74)	73.32 (+15.32)	71.36 (+18.79)	64.56 (+26.75)
CPrompt	88.62	82.02	85.77	76.80	83.97	72.99	77.34	64.80
w/ KAC	89.60 (+0.98)	83.08 (+1.06)	89.04 (+3.27)	80.75 (+3.95)	87.06 (+3.09)	78.54 (+5.55)	85.11 (+7.77)	76.51 (+11.71)

inal hyperparameters to train the model, allowing for a comparison of the differences between classifiers. We implement all compared approaches with their official code and their original selected hyperparameters. For all experiments, we report the average incremental accuracy (the average accuracy over all tasks) and the accuracy of the last task (the overall accuracy after learning the final task).

4.2. Experimental Results

Experiments on ImageNet-R. Tab. 1 compares the accuracies between the baseline methods and those with KAC in the ImageNet-R benchmarks. Replacing the linear classifiers with KAC leads to improvements across all methods, especially in challenging long-sequence scenarios, where gains of 3 to 5 points are observed in most cases. It demonstrates that KAC effectively helps models mitigate forgetting at each step. Furthermore, comparing CODAPrompt and CPrompt, we find that while both perform similarly when using linear classifiers, CODAPrompt outperforms CPrompt when switched to KAC. This indicates that the compatibility of KAC with different methods varies.

Experiments on CUB200. Tab. 2 shows a comparison of the metrics in the CUB200 settings, surprising improvements achieving 10 to 25 percent are observed in long-sequence scenarios. As CUB200 is a fine-grained bird classification dataset, we believe that KAC will perform well with such fine-grained datasets.

Experiments on DomainNet. We conduct experiments on DomainNet for Domain Incremental Learning, aiming to validate the ability of KAC to extend to other continual classification tasks. As shown in Tab. 3, when all approaches are implemented with KAC, the performance achieves an improvement of about 1 percent in average incremental accuracy and about 0.5 percent in last accuracy, demonstrating the robustness of our KAC.

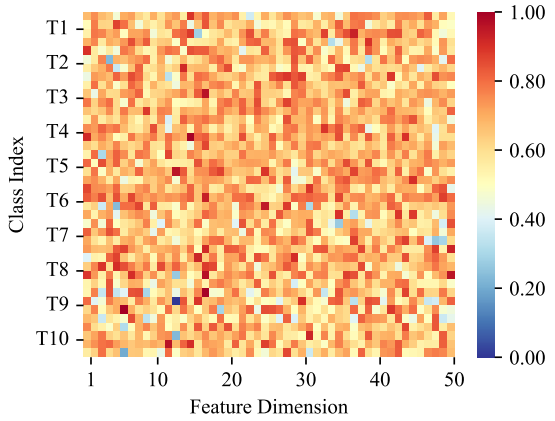


Figure 4. Activation maps for different classes across different channels. The x-axis represents 50 randomly selected channels from feature embeddings, while the y-axis represents classes from different tasks. The colors indicate varying levels of interest.

Visualization of activation maps. Fig. 4 illustrates how different classes activate distinct channels, the differences in activation across different channels for various classes. Only a subset of channels is activated for each category, and updates are applied exclusively to these channels, preventing any impact on the other channels and highlighting the locality advantage in mitigating catastrophic forgetting.

4.3. Ablation Study and Analysis

Ablation on the number of basis functions. The number of basis functions N is a key hyperparameter of KAC. An excessive number of basis functions may lead to additional computations and result in a significantly high dimensionality of W. Conversely, a small value of N may compromise the approximation capacity of KAC. To explore an appropriate value for N, we conduct an ablation study on it. Fig. 5 shows the average incremental accuracy for

Table 3. Results on DomainNet. A Domain Incremental Learning experiment is conducted on it with 6 incremental domains of 345 classes. We report the average incremental accuracy and the accuracy of the last task. The results show that KAC can also work in DIL settings.

Method	Avg	Last
L2P	57.78	49.22
w/ KAC	59.79 (+2.01)	51.10 (+1.88)
DualPrompt	60.96	51.83
w/ KAC	62.06 (+1.10)	52.76 (+0.93)
CODAPrompt	61.61	53.12
w/ KAC	62.78 (+1.17)	53.54 (+0.42)
CPrompt	61.32	52.49
w/ KAC	62.13 (+0.81)	53.02 (+0.53)

Table 4. Ablation study on the structure of the classifier. We replace the spline functions in KAC with MLPs to validate the effectiveness of the KAN structure. Here, w/ MLP represents the MLP trained alongside the model, while w/ MLP (fixed) represents the randomly initialized MLP projection without any updating. The experiments are conducted in the 20 steps ImageNet-R scenario.

	CODAPrompt	w/ KAC	w/ MLP	w/ MLP (fixed)
Avg	80.92	83.59	80.56	65.87
Last	74.40	77.94	73.59	51.03

four approaches using KAC with different numbers of basis functions in the 20 steps experiment on ImageNet-R. The results indicate that simply increasing the number of basis functions does not benefit mitigating forgetting, and further demonstrate that the performance improvement is not due to increasing the dimensionality of the embedding space. Most approaches exhibit better performance when $N = 4$ or $N = 8$, encouraging us to set N as 4 in our experiments.

The KAN structure plays a key role. To demonstrate that the advantages of KAC lie in the introduced KAN structure, not the additional computations, we replace the RBFs with an MLP layer, setting its output dimension to the number of classes and hidden dimension to $N \times n$ to align the number of parameters with KAC using RBFs, allowing us to make a fair comparison between the two structure. Tab. 4 shows the performance of replacing RBFs with the conventional linear classifier with an additional MLP structure implemented on CODAPrompt. Upon comparison, we discover that whether the additional MLP structure is updated alongside the model or not, it does not yield any positive effects. This indicates that the advantages of KAC stem from its KAN structure rather than a simple increase in the dimensionality of the classification space.

Efficiency analysis. In comparison to conventional linear classifiers, our KAC introduces a negligible increase in computational cost and parameter count at the classifier layer. KAC applies fixed Gaussian activation functions to each dimension which almost introduces no extra computations. For a ViT network with an embedding dimension of 768 to classify 100 categories, the additional parameters introduced by KAC amount to only 0.23M, which is negligible compared to 86M parameters of the backbone.

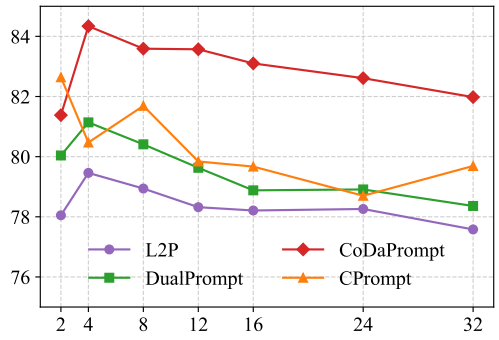


Figure 5. Ablation study on different numbers of basis functions in the 20 steps ImageNet-R scenario. The x-axis represents the number of basis functions, while the y-axis indicates the average incremental accuracy with varying numbers.

5. Conclusions

In this paper, we explore the application of Kolmogorov-Arnold Networks (KAN) in continual learning and propose a novel continual classifier, the Kolmogorov-Arnold Classifier (KAC), which leverages KAN's inherent locality capability to mitigate feature shifts during the learning of new tasks. Our analysis reveals that the limited approximation ability of the B-spline functions in KAN, when applied to high-dimensional data, forces the model backbone to introduce more shifts to accommodate new classes, leading to significant degradation in continual learning performance. This exacerbates the model's forgetting, overshadowing the benefits of locality capability, compared to a traditional linear classifier. To address this issue, we replace the B-spline functions in KAN with Radial Basis Functions (RBFs), which improves performance. KAC demonstrates substantial advantages across various continual learning scenarios, underscoring its effectiveness and robustness. In the future, we plan to explore further possibilities of KAN in continual learning, fully harnessing its inherent strengths.

6. Acknowledgments

This work was funded by NSFC (NO. 62206135, 62225604), Young Elite Scientists Sponsorship Program by CAST (2023QNRC001), "Science and Technology Yongjiang 2035" key technology breakthrough plan project

(2024Z120), Shenzhen Science and Technology Program (JCYJ20240813114237048), and the Fundamental Research Funds for the Central Universities (Nankai Universitiy, 070-63233085). Computation was supported by the Supercomputing Center of Nankai University.

References

[1] Alireza Afzal Aghaei. fkan: Fractional kolmogorov-arnold networks with trainable jacobi basis functions. *arXiv preprint arXiv:2406.07456*, 2024. 3

[2] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 135:38–54, 2021. 1

[3] Zavareh Bozorgasl and Hao Chen. Wav-kan: Wavelet kolmogorov-arnold networks. *arXiv preprint arXiv:2405.12832*, 2024. 3

[4] Roman Bresson, Giannis Nikolentzos, George Panagopoulos, Michail Chatzianastasis, Jun Pang, and Michalis Vazirgiannis. Kagnns: Kolmogorov-arnold networks meet graph learning. *arXiv preprint arXiv:2406.18380*, 2024. 3

[5] Martin Dietrich Buhmann. Radial basis functions. *Acta numerica*, 9:1–38, 2000. 4

[6] Xiuwei Chen and Xiaobin Chang. Dynamic residual classifier for class incremental learning. In *ICCV*, pages 18743–18752, 2023. 1, 2

[7] Gianluca De Carlo, Andrea Mastropietro, and Aris Anagnostopoulos. Kolmogorov-arnold graph neural networks. *arXiv preprint arXiv:2406.18354*, 2024. 3

[8] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022. 1

[9] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pages 86–102. Springer, 2020. 1, 2

[10] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *CVPR*, pages 9285–9295, 2022. 1, 2

[11] David K Duvenaud, Hannes Nickisch, and Carl Rasmussen. Additive gaussian processes. *Advances in neural information processing systems*, 24, 2011. 5

[12] Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999. 1, 3

[13] Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *CVPR*, pages 11483–11493, 2023. 2

[14] Rui Gao and Weiwei Liu. Ddgr: Continual learning with deep diffusion-based generative replay. pages 10744–10763. PMLR, 2023. 2

[15] Zhanxin Gao, Jun Cen, and Xiaobin Chang. Consistent prompting for rehearsal-free continual learning. In *CVPR*, pages 28463–28473, 2024. 3

[16] Zhanxin Gao, Jun Cen, and Xiaobin Chang. Consistent prompting for rehearsal-free continual learning. In *CVPR*, pages 28463–28473, 2024. 1, 2, 6

[17] Remi Genet and Hugo Inzirillo. Tkan: Temporal kolmogorov-arnold networks. *arXiv preprint arXiv:2405.07344*, 2024. 3

[18] Lukasz Golab and M. Tamer Özsu. Issues in data stream management. *SIGMOD Rec.*, 32(2):5–14, 2003. 1

[19] Dipam Goswami, Yuyang Liu, Bartłomiej Twardowski, and Joost van de Weijer. Fecam: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[20] PC Hammer. Adaptive control processes: a guided tour (r. bellman), 1962. 4

[21] Simon Haykin. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1998. 3

[22] Juncai He. On the optimal expressive power of relu dnns and its application in approximation with kolmogorov superposition theorem. *arXiv preprint arXiv:2308.05509*, 2023. 4

[23] Juncai He and Jinchao Xu. Deep neural networks and finite elements of any order on arbitrary dimensions. *arXiv preprint arXiv:2312.14276*, 2023. 4

[24] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021. 6

[25] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989. 2, 3

[26] Amanda A Howard, Bruno Jacob, Sarah H Murphy, Alexander Heinlein, and Panos Stinis. Finite basis kolmogorov-arnold networks: domain decomposition for data-driven and physics-informed problems. *arXiv preprint arXiv:2406.19662*, 2024. 3

[27] Yusong Hu, Zichen Liang, Xialei Liu, Qibin Hou, and Ming-Ming Cheng. Reformulating classification as image-class matching for class incremental learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(1):811–822, 2025. 2

[28] Kishaan Jeeveswaran, Prashant Shivaram Bhat, Bahram Zonooz, and Elahe Arani. BiRT: Bio-inspired replay in vision transformers for continual learning. pages 14817–14835, 2023. 2

[29] Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. In *CVPR*, pages 16071–16080, 2022. 2

[30] Junsu Kim, Hoseong Cho, Jihyeon Kim, Yihalem Yimolal Tiruneh, and Seungryul Baek. Sddgr: Stable diffusion-based deep generative replay for class incremental object detection. In *CVPR*, pages 28772–28781, 2024. 2

[31] Taehoon Kim, Jaeyoo Park, and Bohyung Han. Cross-class feature augmentation for class incremental learning. In *AAAI*, pages 13168–13176, 2024. 1, 4

[32] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 2

[33] Benjamin C Koenig, Suyong Kim, and Sili Deng. Kan-odes: Kolmogorov–arnold network ordinary differential equations for learning dynamical systems and hidden physics. *Computer Methods in Applied Mechanics and Engineering*, 432: 117397, 2024. 3

[34] Andreĭ Nikolaevich Kolmogorov. *On the representation of continuous functions of several variables by superpositions of continuous functions of a smaller number of variables*. American Mathematical Society, 1961. 2, 3

[35] Mario Köppen. On the training of a kolmogorov network. In *Artificial Neural Networks—ICANN 2002: International Conference Madrid, Spain, August 28–30, 2002 Proceedings 12*, pages 474–479. Springer, 2002. 4

[36] Ming-Jun Lai and Zhaiming Shen. The kolmogorov superposition theorem can break the curse of dimensionality when approximating high dimensional functions. *arXiv preprint arXiv:2112.09963*, 2021. 4

[37] Ziyao Li. Kolmogorov-arnold networks are radial basis function networks. *arXiv preprint arXiv:2405.06721*, 2024. 3, 4, 5

[38] Zhizhong Li and Derek Hoiem. Learning without forgetting. *PAMI*, 40(12):2935–2947, 2017. 4

[39] Henry W Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168:1223–1247, 2017. 4

[40] Xialei Liu, Yu-Song Hu, Xu-Sheng Cao, Andrew D Bagdanov, Ke Li, and Ming-Ming Cheng. Long-tailed class incremental learning. In *ECCV*, pages 495–512. Springer, 2022. 2

[41] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024. 2, 3, 4

[42] R. O. Malashin and M. A. Mikhalkova. Avoiding catastrophic forgetting via neuronal decay. In *2024 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF)*, pages 1–6, 2024. 3

[43] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, pages 109–165. Elsevier, 1989. 2

[44] Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. 36, 2024. 1, 2, 4

[45] Hadrien Montanelli and Haizhao Yang. Error bounds for deep relu networks using the kolmogorov–arnold superposition theorem. *Neural Networks*, 129:1–6, 2020. 4

[46] Mark JL Orr et al. Introduction to radial basis function networks, 1996. 4

[47] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019. 1

[48] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019. 6

[49] Kaihuai Qin. General matrix representations for b-splines. In *Proceedings Pacific Graphics' 98. Sixth Pacific Conference on Computer Graphics and Applications (Cat. No. 98EX208)*, pages 37–43. IEEE, 1998. 3

[50] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 1, 2

[51] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *NeauIPS*, 30, 2017. 2

[52] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *CVPR*, pages 11909–11919, 2023. 1, 2, 3, 6

[53] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 6

[54] Fu-Yun Wang, Da-Wei Zhou, Liu Liu, Han-Jia Ye, Yatao Bian, De-Chuan Zhan, and Peilin Zhao. Beef: Bi-compatible class-incremental learning via energy-based expansion and fusion. In *ICLR*, 2022. 2

[55] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. 36, 2024. 3

[56] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. 35:5682–5695, 2022. 2, 3, 6

[57] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *ECCV*, pages 631–648. Springer, 2022. 1, 3, 6

[58] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *CVPR*, pages 139–149, 2022. 1, 3, 6

[59] Haitao Wen, Lili Pan, Yu Dai, Heqian Qiu, Lanxiao Wang, Qingbo Wu, and Hongliang Li. Class incremental learning with multi-teacher distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28443–28452, 2024. 1, 2, 4

[60] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *CVPR*, pages 3014–3023, 2021. 1

[61] Guanglei Yang, Enrico Fini, Dan Xu, Paolo Rota, Mingli Ding, Moin Nabi, Xavier Alameda-Pineda, and Elisa Ricci. Uncertainty-aware contrastive distillation for incremental semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2567–2581, 2022. 2, 4

[62] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *CVPR*, 2020. 1, 4

[63] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *CVPR*, pages 19148–19158, 2023. 1, 2

[64] Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *CVPR*, pages 23554–23564, 2024. 2

[65] Huiping Zhuang, Zhenyu Weng, Run He, Zhiping Lin, and Ziqian Zeng. Gkeal: Gaussian kernel embedded analytic learning for few-shot class incremental task. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7746–7755, 2023. 4