Kolmogorov Arnold Networks for Class Incremental Learning

Shashwat Roy (B21CS071) , Sukriti Goyal (B21CS075)

Abstract

Kolmogorov-Arnold Networks (KANs) present a unique approach in the area of Deep Learning. Proposed in 2024, it offers certain advantages in comparison to Multi-Layer Perceptrons such as having more interpretability and faster neural scaling laws. In this report, we explore the applications of KANs in the domain of class-incremental learning. We also present our findings and inferences for the same.

1. Introduction

1.1. Class-Incremental Learning

Class-Incremental Learning (CIL) is a machine learning paradigm designed to enable models to learn and adapt continually in scenarios where new classes of data emerge over time. Unlike traditional training, which assumes access to a fixed and complete dataset, CIL deals with evolving data streams where only new class instances are available at each stage. The goal is to build a universal classifier capable of recognizing all previously seen and newly added classes, while addressing the critical challenge of catastrophic forgetting—a phenomenon where learning new information leads to the loss of knowledge about previously learned classes. CIL is particularly important for applications requiring lifelong learning, such as autonomous systems and adaptive AI, where data is dynamic and cannot be stored indefinitely due to storage constraints or privacy concerns.

1.2. Kolmogorov-Arnold Network

Kolmogorov-Arnold Networks are inspired by the Kolmogorov-Arnold representation theorem. The original version of the Kolmogorov-Arnold representation theorem states that for any continuous function $f : [0, 1]^d \to \mathbb{R}$, there exist univariate continuous functions $g_q, \psi_{p,q}$ such that

$$f(x_1, \ldots, x_d) = \sum_{q=0}^{2d} \Phi \left(\sum_{p=1}^{d} \phi_{p,q}(x_p) \right). \qquad (1)$$

where $\Phi : \mathbb{R} \to \mathbb{R}$ and $\psi_{p,q} : [0, 1] \to \mathbb{R}$.

This means that $(2d + 1)(d + 1)$ univariate functions Φ and $\phi_{p,q}$ are sufficient to exactly represent a d-variate function. The corresponding network for this equation would appear as a two-layer neural network with activation functions placed on edges instead of nodes (simple summation is performed on nodes), and with width $2d+1$ in the middle layer. However, this network is too shallow to approximate any function well in practice. Thus, the authors of the paper extend this concept to propose a deeper Kolmogorov-Arnold network.

1.3. Why KAN?

In KANs, the activation function $\phi(x)$ is the sum of the basis function $b(x)$ and the spline function. The spline function spline(x) is parametrized as a linear combination of B-splines where the corresponding coefficients are learnable. In the domain of class incremental learning, the distribution of incoming data (new classes) changes over time. The authors of the paper demonstrate the possibility of KANs overcoming the issue of catastrophic forgetting that is notorious in MLPs. They suggest that, due to the splines local plasticity, KANs can avoid catastrophic forgetting. The spline bases have local control , i.e. a sample will only affect the nearby local spline coefficients. This is not the case with MLPs which use global activation functions like ReLU and Tanh.

2. Our Approach

For the task of image classification, we considered some popular architectures such as AlexNet and VGG16. In such CNN models, the convolutional layers are expected to extract the meaningful features in the input images. The extracted embeddings are then forwarded to the fully connected layers which classify the input accordingly. In class-incremental learning, the data distribution changes over time. Thus, we decided to replace the second last fully connected in the aforementioned architectures. We assume that this drift in data distribution will also be reflected in the embeddings from the convolutional layers. Thus, if we use KAN layers to analyse these embeddings, we may mitigate the issue of catastrophic forgetting. We also noticed no significant improvement in performance when using multiple KAN layers which is why we prefer replacing only one fully

connected layer with a KAN layer.

2.1. Class-incremental algorithms used

2.1.1. LwF

Learning without forgetting is a class incremental algorithm which can be seen as a combination of distillation networks and fine-tuning. It uses only the new data as it assumes that the past data is unavailable. The algorithm first records the responses for the old tasks from the original network on the new task images. During training, the network's current predictions for the old tasks are compared to the recorded outputs using the Knowledge Distillation loss ($\mathcal{L}_{\text{old}}$). The new task loss ($\mathcal{L}_{\text{new}}$) is calculated using multinomial logistic loss which encourages the network to learn the new task effectively by minimizing the prediction error between the network's outputs for the new task and the ground truth labels from the new task's dataset.

$$\theta_s^*, \theta_o^*, \theta_n^* \leftarrow \arg\min_{\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n} \Big(\lambda_0 \mathcal{L}_{\text{old}}(Y_o, \hat{Y}_o) + \mathcal{L}_{\text{new}}(Y_n, \hat{Y}_n) + \mathcal{R}(\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n) \Big)$$

Where:

- θ_s are the shared parameters
- θ_o are the old task parameters
- θ_n are the new task parameters
- λ_o is a loss balance weight
- $\mathcal{R}$ denotes regularization using weight decay

2.1.2. iCaRL

Incremental Classifier and Representation Learning (iCaRL) is a class-incremental algorithm that learns both classifiers and a feature representation simultaneously. It uses a nearest-mean-of-exemplars strategy for classification and maintains a fixed-size memory of exemplars to prevent forgetting old classes while learning new ones.

$$y^* = \arg\min_{y=1,\ldots,t} \|\varphi(x) - \mu_y\|,$$

where:

- $\mu_y = \frac{1}{|P_y|} \sum_{p \in P_y} \varphi(p)$ is the mean feature vector (prototype) of class y,
- $\varphi(x)$ is the feature vector of input x, and
- P_y is the set of exemplars for class y.

During training, iCaRL processes new classes incrementally. For each new class:

- It constructs exemplar sets by selecting m samples that best approximate the class mean in feature space.
- It reduces exemplar sets of old classes to ensure the total memory size K is maintained.

2.2. Experimental setup

We have used the same datasets as the ones that were used for evaluation in papers introducing iCarl and LwF. Due to compute constraints, we use a subset of CUB200 dataset (100 classes), CIFAR100 and MIT Scenes. We train AlexNet and VGG16 models for both instances (MLP and KAN-replaced). We also repeat these experiments on pre-trained models (ImageNet) for the LwF algorithm. The code implementations were refered from: (Zhou et al., 2023)

3. Insights

Initially, we replaced the first two fully connected layers with KAN layers and compared the performances on the CUB200 dataset. We found that the performances of the model on replacing only the second-last MLP layer with a KAN layer exceeded that of the original model and that of the model with 2 MLP layers swapped with corresponding KAN layers.

We then repeated these experiments with AlexNet and VGG16 models having pre-trained ImageNet weights from pytorch. It was found that when the models were trained for more epochs, there was a higher improvement in accuracy for the KAN-replaced models than in the original models. The KAN-replaced models outperformed the original models on the CUB200 dataset. However, the original models outperformed the latter on the MIT Scenes dataset.

Based on our observation from the results, we could infer that using a KAN Linear layer as a second last layer instead of the last layer gave us better accuracy values for Learning with Forgetting regime of Class Incremental Learning. This is so because KANs are better at continual learning as it prevents catastrophic forgetting as mentioned in detail in Section 3.5 of the research paper introducing KANs(Liu et al., 2024).

4. Future Work

KANs seemed to be slightly better at MLPs as can be seen from the results. So further on we aim to experiment the effectiveness of KAN on other class incremental learning algorithms such as prompt-based class incremental learning algorithms. We also aim to test the impact of convolutional KAN layers on such model architectures in different CIL

Model	Epoch	Learning Rate	Decay	1-Layer KAN	2-Layer KAN	MLP
AlexNet	400	0.001	0.1	24.24	23.128	23.378
AlexNet	400	0.01	0.005	22.57	23.27	-
AlexNet	400	0.05	0.01	23.699	24.58	23.34
AlexNet	200	0.0005	0.1	22.16	-	21.8
AlexNet	200	0.0001	0.1	14.895	-	17.49
AlexNet	200	0.01	0.5	24.044	20.029	23.838

Table 1. Results of LwF for experiments with AlexNet on CUB200 dataset (100 classes).

Learning Rate	1-Layer KAN	MLP
0.001	31.927	32.884
0.01	25.766	26.613

Table 2. Results of LwF for experiments with AlexNet architecture on CIFAR100 dataset

Model	Epoch	Learning Rate	Decay	1-Layer KAN	MLP
AlexNet	50/75	0.001	0.1	32.763	34.57
AlexNet	100/150	0.001	0.1	35.81	34.7665
AlexNet	100/150	0.01	0.001	30.483	29.89
VGG16	75/100	0.001	0.1	38.78	37.87
VGG16	75/100	0.001	0.1	12.63	12.47
VGG16	75/100	0.001	0.1	41.21	38.98

Table 3. Results of LwF for experiments on CUB200 dataset (100 classes) with AlexNet and VGG16 pre-trained on ImageNet

Model	Epoch	Learning Rate	Decay	1-Layer KAN	MLP
AlexNet	150	0.001	0.1	15.46	21.49
AlexNet	250	0.005	0.01	38.2	41.6
VGG16	50/75	0.005	0.01	48.55	50.285
VGG16	50/75	0.001	0.1	64.77	66.33
VGG16	100/150	0.001	0.1	69.79	70.1
VGG16	250	0.001	0.1	68.18	56.76

Table 4. Results of LwF with AlexNet and VGG16 architectures pre-trained on ImageNet for experiments with the Scenes dataset

Epochs	MLP		KAN	
	CNN	NME	CNN	NME
50	63.31	66.88	63.49	69.25
50	65.28	69.25	64.824	69.74
100	-	-	62.5	65.7

Table 5. Results of iCarl with AlexNet architecture on CIFAR100 with learning rate = 0.001 and weight decay = 0.001

algorithms.

References

Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T. Y., and Tegmark, M. Kan: Kolmogorov-arnold networks. 2024. URL https://arxiv.org/abs/2404.19756.

Zhou, D.-W., Wang, F.-Y., Ye, H.-J., and Zhan, D.-C. Pycil: a python toolbox for class-incremental learning. *SCIENCE CHINA Information Sciences*, 66(9):197101, 2023. doi: https://doi.org/10.1007/s11432-022-3600-y. URL https://github.com/G-U-N/PyCIL.