Evaluating Federated Kolmogorov-Arnold Networks on Non-IID Data

Arthur M. Sasse
Universidade Federal do Rio de Janeiro
artsasse@cos.ufrj.br

Claudio M. de Farias
Universidade Federal do Rio de Janeiro
cmicelifarias@cos.ufrj.br

Abstract

Federated Kolmogorov-Arnold Networks (F-KANs) have already been proposed, but their assessment is at an initial stage. We present a comparison between KANs (using B-splines and Radial Basis Functions as activation functions) and Multi-Layer Perceptrons (MLPs) with a similar number of parameters for 100 rounds of federated learning in the MNIST classification task using non-IID partitions with 100 clients. After 15 trials for each model, we show that the best accuracies achieved by MLPs can be achieved by Spline-KANs in half of the time (in rounds), with just a moderate increase in computing time.

1 Introduction

Since the paper by [21], many works have investigated Federated Learning potential as a privacy-focused alternative to centralized Machine Learning (ML) models, offering the possibility of collaborative ML without sharing users' data, just models. Seven years later, there are still challenges to the real-life application of federated learning, like guaranteeing high accuracies across clients with non-iid data and reducing communication costs between clients and servers as described by [20].

In 2024, [19] proposed a new kind of neural network that could help tackle these problems. The Kolmogorov-Arnold Networks (KANs) promise higher accuracy with fewer parameters than Multi-Layer Perceptrons, the traditional neural networks. Having a completely different structure, inverting the roles of edges and nodes in a neural network, and built with learnable activation functions, KANs also offer the possibility of creating more interpretable and lean models [19], which in turn could help scientists discover even symbolic formulas for their tasks. The paper by [19] has not yet been officially published or peer-reviewed, but since it's been made available on Arxiv[1], about 350 studies have been produced exploring the possibilities opened by this new paradigm in ML, according to Google Scholar.

KANs have already been combined with other relevant concepts in ML, like convolutional models by [5], or graph neural networks by [14]. Similarly, [28] and [7] initiated studies exploring KANs in federated learning. The potential for achieving higher accuracies with fewer parameters than MLPs and avoiding catastrophic forgetting, as described by [19], could be useful for handling communication restraints and heterogeneous data across clients, respectively. However, there is no evidence to guarantee the convergence rate of F-KANs, and the few experimental evaluations publicly available do not explicitly address the non-iid problem, nor do they compare the F-KANs against federated MLPs with a similar number of parameters. Also, initial empirical evaluation of KANs against MLPs suggests that the execution times of the former could be ten times larger than those of the latter, threatening its use in practice.

This work aims to fill the gaps described above by conducting simulations with federated learning for KANs and MLPs under experimental conditions that are more similar to typical studies in FL.

---

[1]https://arxiv.org/abs/2408.10205

This study is a fundamental step in deciding whether we can integrate KANs into the current FL ecosystem, which specific adaptations they may need due to their structural differences from the classical MLPs, and if KANs can be considered valuable tools for solving FL practical problems. Our focus is evaluating whether KANs can also achieve higher accuracies than MLPs in federated learning and the relationship between both models' computing times.

First, we compare KANs and MLPs (with a similar number of parameters) concerning the global model test accuracy achieved over 100 rounds of FL in the MNIST digits classification task. The dataset is partitioned over 100 clients in a "pathological" non-iid way, with each client having data for only two labels. For example, one client has data for some of the handwritten digits "1" and "2", while another has part of the data from digits "7" and "8". For KANs, we test two kinds of activation functions: B-Splines, as initially proposed by [19], and Radial Basis Functions (RBFs), as proposed by [17]. We use FedAvg with momentum as the aggregation algorithm to achieve provable convergence for the non-iid data, as described by [10].

After 100 rounds of communication, we show, with statistical significance, that Spline-KANs achieve higher accuracies than MLPs during all phases of federated learning, particularly during the first half of the simulation. RBF-KANs perform poorly compared to the other models during most rounds, suggesting we may need to adjust the aggregation algorithm according to each KAN activation function. The execution times from the KAN models are indeed higher than the MLPs' times, as expected, but much lower than initially reported in earlier studies, probably due to the new and more efficient implementations.

The document is divided as follows: in Section 2, we compare this work with others in the field of federated KANs; in Section 3, we describe our experimental setup and statistical tests; in Section 4 we present graphs and the results of our tests, discussing our findings; in Section 5 we comment on the limitations of our study; and in Section 6 we synthesize our findings and perspectives about federated KANs.

2   Related Works

[19] introduced the Kolmogorov-Arnold Networks, an alternative to MLPs, inspired by the Kolmogorov-Arnold representation theorem. KANs are also multi-layer fully-connected networks, however, their activation functions are learnable and placed on the edges of the network, while typical MLPs have fixed activation functions placed on its nodes. KANs, in turn, simply sum the activation values on its nodes. The linear weights of the MLPs are substituted by splines, 1D non-linear functions, whose parameters can be learned during training. An illustration of a KAN is shown in Figure 1. The authors argue that KAN's strength relies on mixing the qualities of MLPs and splines: capturing the compositional (additive) structure of data and bringing more accuracy to the approximation of univariate functions, respectively.
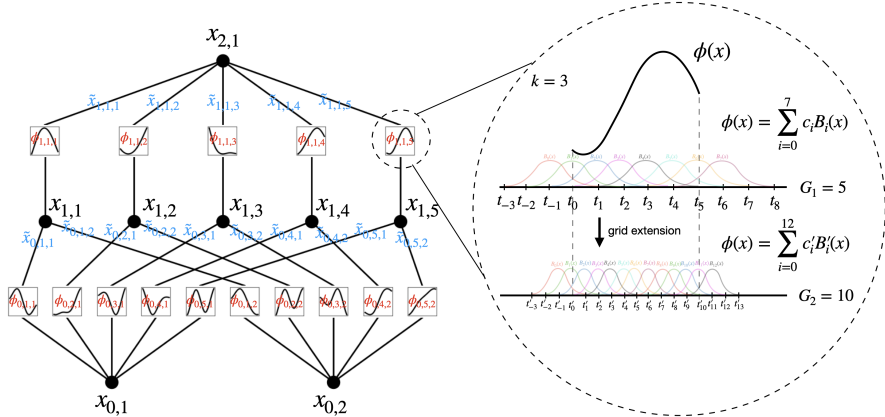

Figure 1: The external and internal structures of KANs. Reproduced from Liu et al. (2024) [19].

2

[19] presented the theory behind these networks and provided the Python package *pykan*[2] for practical application. They conducted empirical evaluations with physical and mathematical problems having small associated datasets, in centralized settings. Our study focuses on a federated setup with a much larger dataset (MNIST), which is classically used for model evaluation in the Machine Learning area. We also used novel implementations of KAN, focused on more efficiency, which will be described further in this article.

[17] presented a new type of KAN, using radial basis functions as activations functions, and made the implementation available in GitHub as a package for Python called FastKAN. The author claims that FastKAN is faster than the traditional B-Spline implementation based on an evaluation with a centralized MNIST classification task. We use FastKAN to classify our pathologically partitioned MNIST dataset.

[28] coined the term "F-KAN", presenting the method for federating KANs, based on the FedAvg algorithm for MLPs. The authors evaluated the proposed model through a classification task using the Iris dataset [12]. The Iris dataset consists of data for 50 samples of flowers from three different species, described by four features. The dataset is partitioned between two clients in an iid way. The simulation comprised one execution for 20 rounds of federated learning, comparing an MLP and a Spline-KAN model, each with three hidden layers of 20 neurons. The KAN model showed superior performance for four metrics: accuracy, recall, precision, and F1-score. This work also compares KANs and MLPs in a federated setting. However, we change some parameters in trying to simulate a more realistic FL configuration: instead of the Iris Dataset, we use MNIST, which comprises 70,000 examples (60,000 for training and 10,000 for testing) from 10 classes of handwritten digits, represented by images with 28*28 pixels (784 features); we divide the data over 100 clients in a non-iid manner, and only 10% of clients participate in each round, simulating clients that are not always available; we run 15 executions for each model to achieve more statistically significant results; we compare MLPs not only against Spline-KANs but also against RBF-KANs; we run 100 communication rounds to capture more of each model behavior through time; and we reduce the number of KAN neurons so both models have a similar number of parameters. Table 1 shows the detailed comparison between each study.

[7] present a study in progress about federated learning with KANs, which use different kinds of wavelets as their activation functions, based on [6]. The authors show partial results from a federated Mexican hat Wavelet-KAN in a [28*28, 64, 10] layer arrangement for the MNIST dataset classification. They claim Wavelet-KANs are superior to Spline-KANs. However, direct comparisons between models were not available at the time of this writing. The results are averaged over a non-disclosed number of trials for each number of clients (10, 20, and 50) and also for a centralized model. The authors specify that each model is trained over 50 epochs, but we could not find how many communication rounds happened in each simulation. Besides differences in the experimental setups, our paper is focused only on Spline-KANs and RBF-KANs, not including Wavelet-KANs. The Wavelet-KAN implementation is available in GitHub but not as a Python package, which created some difficulties adapting it to our study, which we hope to overcome in future studies.

Our work differentiates itself by presenting a broader set of experiments on federated KANs, drawing our conclusions from more common settings in the ML and FL areas. We believe this work can serve as a baseline for future experiments comparing KANs and MLPs' capacities in federated architectures.

3   Methodology

This work is guided by two research questions and their respective metrics and hypotheses.

- **Research Question 1**
  - **Question**: Do KANs achieve higher test accuracies than MLPs across 100 rounds of federated learning in the MNIST classification task?
  - **Metric**: Test accuracy of the global models for rounds [10, 20, 30, 40, 50, 60, 70, 80, 90, 100].
  - **Null Hypothesis**: KANs achieve a test accuracy lower than or equal to that of MLPs in a given round of federated learning.

---

[2]`https://github.com/KindXiaoming/pykan`

- **Alternative Hypothesis**: KANs achieve a higher test accuracy than MLPs in a given round of federated learning.
- **Hypothesis Test**: Welch's t-test [26], since we can't assume the models' accuracies will have the same variances, and it's a robust test against deviations from normality [22].

- **Research Question 2**
    - **Question**: What is the relationship between the execution times of KANs and MLPs for 100 rounds of federated learning in the MNIST task?
    - **Metric**: Execution time, in seconds.
    - **Method**: Calculate the confidence interval for the ratio between the execution times of KANs and MLPs, using bootstrapping [11] (10,000 bootstrapped samples of size 15).

3.1 Experimental Setup

Table 1: Comparison of experimental setups between [28] and our work.

Parameter	[28]	Our Work
General		
Dataset	Iris	MNIST
Classes	3	10
Features	4	28 * 28
Samples	150	60,000 + 10,000
Clients	2	100
Partition	iid	non-iid
Rounds	20	100
Batch size	16	64
Local Epochs	20	5
Learning rate	0.001	0.1
Clients per round	100%	10%
Weight decay	1×10^{-5}	–
Dropout probability	0.5	–
Aggregation	FedAvg	FedAvg
Optimizer	Adam	SGD with momentum = 0.9
KAN Activation	B-Spline	B-Spline, RBF
Executions per model	1	15
KAN		
Total Parameters (Spline)	12,383	196,320
Total Parameters (RBF)	–	178,410
Hidden layer sizes	[20, 20, 20]	[24, 24]
Grid size	5	5
Spline order	3	3
RBF Centers	–	8
MLP		
Total parameters	1,003	199,210
Hidden layer sizes	[20, 20, 20]	[200, 200]

To make a fair comparison between KANs and MLPs, we defined a maximum number of parameters for each model. First, we fix the architecture of the MLP classifier, inspired by one of the models studied on the FedAvg paper by [21], a [28*28, 200, 200, 10] neural network. Based on that, we built a Spline-KAN model with the same number of layers, calculating the number of neurons needed in each layer to achieve, at most, the same number of parameters used by the MLP, resulting in a [28*28, 24, 24, 10] network. We repeated this architecture for the RBF-KAN. Table 1 details the resulting architectures and hyperparameters.

We implemented our simulation using PyTorch ([2]) and the Flower ([3]) framework for federated learning. We used the efficient-kan implementation ([4]) of Spline-KANs, whose contribution to enhancing KAN's performance has already been acknowledged by the authors of the original KAN paper in a subsequent article ([18]). For RBF-KANs, we used [17] implementation. Our code repository is available at `https://github.com/artsasse/fedkan`.

All simulations were run in the Google Colaboratory [3] environment, with an Intel Xeon CPU @ 2.20 GHz with two virtual CPUs and 13 GB RAM, alongside a NVIDIA Tesla T4 GPU with 15GB VRAM.

The local models were aggregated by the server using the FedAvg algorithm ([21]) with momentum, since [10] proved how using momentum can make FedAvg converge without needing to use decaying learning rates, thus simplifying our setup.

The MNIST ([15]) is a handwritten digits dataset commonly used for evaluating classification models. We used its original split, with 60,000 samples for training and 10,000 for testing. We kept the test samples together for centralized evaluation of the global model at the end of each round of FL. The training set was "pathologically" partitioned between 100 clients so that each client holds examples for only two digits, guaranteeing the non-iid property, a common challenge for real-life FL applications. Figure 2 shows graphically how the data was partitioned. Overall, the partitions are slightly balanced: the number of examples for each client varies between 488 and 792, with mean = 600. This skewed division was inspired by [21] and [16]. However, the former used balanced partitions, and the latter created unbalanced partitions following a power law.
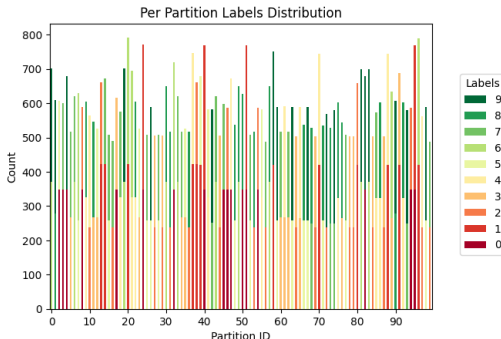

Figure 2: Per partition labels distribution.

Since we only select a fraction of the clients for training in each round of FL, the training accuracy is calculated using different data for each round of communication. Thus, its values could change dramatically from one round to the next even if the global model stayed the same.

4 Results

To answer our **Research Question 1**, we gathered the test accuracy metric, represented in Figure 3, for each of the 100 rounds of FL. Intuitively, the Spline-KAN appears to have the overall best performance, followed by the MLP, while the RBF-KAN shows very low accuracy, even in the final rounds of communication. In Figure 4, we also represent the training accuracy (4a), the test loss (4b), and the training loss (4c), which show tendencies aligned with what we observe for the test accuracy.

We also present the test accuracies for ten different rounds in Table 2, showing the mean and standard error over 15 trials for each model. The results of the actual hypotheses tests are represented further in Table 3, where we present the p-values for the Welch's t-test [26]. Comparing Spline-KANs and MLPs, for a 90% significance level, we can refute the null hypothesis for all the selected rounds. For a 95% significance level, we refute the null hypothesis from round 10 to 70. The differences between mean test accuracies are greater than 20% from round 10 to round 60. Besides the statistically significant superiority of Spline-KANs during all the selected rounds, we note that the Spline-KAN's test accuracy increases faster during the first half of the simulation, slowing down to almost a linear
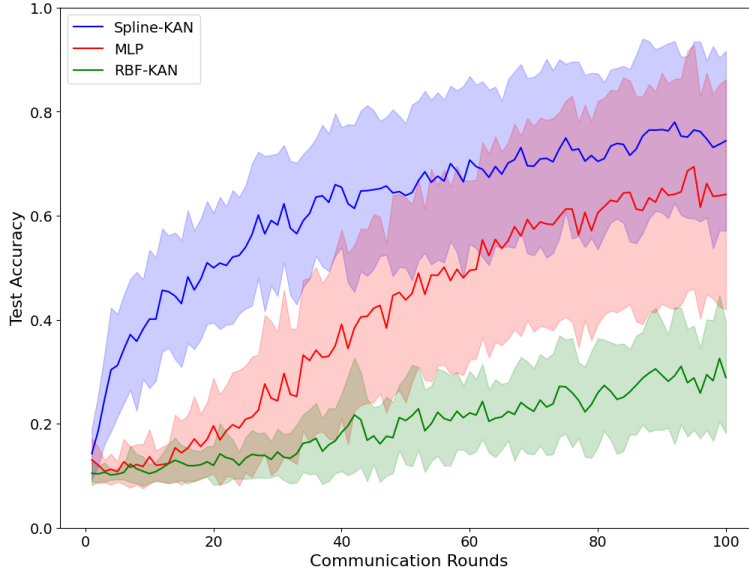
---

[3]https://colab.google

Figure 3: Mean test accuracy for 100 rounds, with error bands.

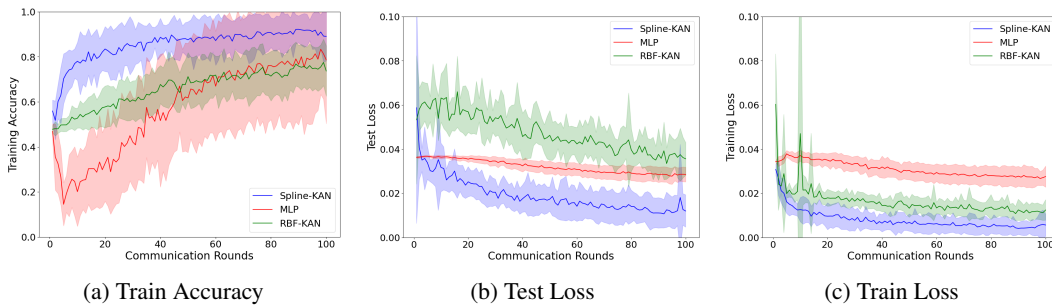(a) Train Accuracy   (b) Test Loss   (c) Train Loss

Figure 4: Mean accuracy and loss metrics for 100 rounds, with error bands.

increase for the second half. The MLPs presented a practically linear increase during the 100 rounds. On average, the best accuracies are achieved by the MLPs during the last two rounds, while similar values had already been achieved by Spline-KANs in round 40. The RBF-KANs also showed a linear-like increase in its metric, but much slower than the MLPs, reaching a maximum average accuracy a little higher than 0.2. We conjecture that RBF-KANs may need other aggregation algorithms, more adapted to its parameters than FedAvg, to reach good accuracies, which should be investigated in future works.

First of all, our results are positive evidence for using Spline-KANs in FL, since they outperformed, during all the simulation phases, the MLPs, which are the original target of FedAvg and the other FL tools used in this work. Also, the accelerated increase in accuracy in early rounds could be an interesting feature for federated systems that do not prioritize a very high accuracy but aim to minimize the training rounds needed to reach a good enough generalization. This limitation could be due to communication or battery restraints, common for sensor networks and the Internet of Things [20]. The model compressibility and energy usage are properties that should be further studied when considering the application of KANs in these domains [20].

Regarding **Research Question 2**, we can observe the analysis results in Table 4. We can see that RBF-KANs have execution times similar to those of MLPs. And, even though Spline-KANs have greater variance in their execution times and can be almost $1.36\times$ slower than MLPs, that difference is not as dramatic as expected. In [19] and [28], the authors related that KANs had taken 10x or even 100x longer than MLPs to execute. We note that both works reported results from the initial implementation of KANs. In our work, we use new implementations of KAN, [4] for Spline-KAN,

and [17] for RBF-KAN, built specially to tackle these performance issues. Moreover, let's consider cases where we want the global model to achieve a specific accuracy. The Spline-KANs may reach that target much earlier than MLPs regarding communication rounds, which would translate to an earlier stoppage in execution and a shorter computing time. In our case, if our goal were a 65% accuracy, MLPs would have stopped by round 90 and Spline-KANs by round 40, on average.

Table 2: **Test accuracy over communication rounds**. Bold font is used to indicate when each model achieved a 0.652 accuracy, which is the best value obtained by the MLP. Spline-KANs reach the same accuracy in less than half of the rounds needed by the MLP.

Round	Spline-KAN Mean $\pm$ Std Error		RBF-KAN Mean $\pm$ Std Error		MLP Mean $\pm$ Std Error	
10	0.401 $\pm$	0.138	0.104 $\pm$	0.018	0.137 $\pm$	0.054
20	0.500 $\pm$	0.109	0.120 $\pm$	0.039	0.196 $\pm$	0.095
30	0.582 $\pm$	0.142	0.146 $\pm$	0.046	0.244 $\pm$	0.137
40	**0.654** $\pm$	0.119	0.183 $\pm$	0.077	0.391 $\pm$	0.188
50	0.639 $\pm$	0.144	0.206 $\pm$	0.087	0.438 $\pm$	0.204
60	0.707 $\pm$	0.167	0.221 $\pm$	0.083	0.495 $\pm$	0.178
70	0.695 $\pm$	0.177	0.224 $\pm$	0.085	0.574 $\pm$	0.204
80	0.704 $\pm$	0.175	0.259 $\pm$	0.108	0.606 $\pm$	0.222
90	0.765 $\pm$	0.165	0.293 $\pm$	0.109	**0.652** $\pm$	0.229
100	0.744 $\pm$	0.179	0.289 $\pm$	0.111	0.641 $\pm$	0.229

Table 3: **Mean accuracy differences and p-values over communication rounds for Spline-KAN compared to MLP.** Bold font in the middle column indicates rounds in which the mean accuracy difference between Spline-KANs and MLPs was greater than 20%. Bold font in the right column indicates rounds in which we refuted the null hypothesis with a 95% significance level, meaning the Spline-KAN's accuracy was statistically superior.

Round	$\text{Acc}_{\text{Spline}} - \text{Acc}_{\text{MLP}}$	$\text{Acc}_{\text{Spline}} > \text{Acc}_{\text{MLP}}$ (p-value)
10	**0.265**	**0.0000**
20	**0.304**	**0.0000**
30	**0.338**	**0.0000**
40	**0.263**	**0.0001**
50	**0.201**	**0.0023**
60	**0.212**	**0.0011**
70	0.121	**0.0474**
80	0.099	0.0940
90	0.113	0.0669
100	0.103	0.0905

Table 4: **Execution time analysis for different models over 100 rounds**.

Model	Mean Time (s)	Std Error	Ratio (95% CI)
MLP	3593	124	$1.00\times$
Spline-KAN	4192	1183	$1.02 \times -1.36\times$
RBF-KAN	3640	114	$0.99 \times -1.04\times$

5 Limitations

A more realistic way to assess federated learning concerning data partitioning would have been to utilize the FEMNIST (Federated Extended MNIST) dataset [9]. Instead of creating an artificial posthoc partition, FEMNIST gives each client access to data from just one handwritten digit author, generating naturally non-i.i.d partitions. In the future, we plan to evaluate F-KANs on FEMNIST and other tasks besides digit classification.

Convolutional models are generally considered a better option for classifying images than fully connected networks [13]. We are unsure if using convolutional networks would cause the disparity between MLPs and KANs to grow or shrink. Even though convolutional KANs have already been proposed [5], because these works are relatively recent, we believe potential flaws in these early implementations could skew the results and jeopardize the study's validity.

There is still no standard method for comparing KANs and MLPs for the same tasks. In an effort to provide a fair comparison in terms of each model's generalization capacity, we decided to utilize a similar number of parameters. On the other hand, arbitrary choices made about the construction of both models could have affected the outcomes in unknown ways. In future works, we intend to perform a hyperparameter optimization to make a fairer comparison between each type of neural network. Also, there are still other activation functions available for KANs that were not investigated in this work, like wavelets [8], ReLU [23], Chebyshev [25], Jacobi [24] [1] and Fourier [27] functions.

Since MLPs have been used much longer than KANs, many specific software and hardware optimizations not used in this work might drastically reduce the MLP execution times. However, if KANs become a popular model for the machine learning community, new optimizations and performance tricks are expected to be created for them, too.

6 Conclusion

In this work, we compared two types of KANs against an MLP model over 100 rounds of federated learning for image classification. We focused on the test accuracy for many simulation phases and the total execution time. Spline-KANs performed better than MLPs during all the simulation phases, especially during the first 60 rounds, showcasing how these types of KANs could be an alternative to achieve higher accuracies in fewer rounds of FL. RBF-KANs reached accuracies well below those of MLPs, demonstrating how the KAN activation function may strongly impact its performance. Both KAN implementations observed execution times at most $1.36\times$ the MLP's execution time, which is way faster than initial implementations of KAN achieved in centralized and federated simulations. Before obtaining a theoretical explanation for the differences between each model, it is impossible to estimate the generality of these results for different FL configurations. Regardless, this highly significant result indicates that KANs could be an interesting path to solve some of the most relevant problems in federated learning.

References

[1] Alireza Afzal Aghaei. fkan: Fractional kolmogorov-arnold networks with trainable jacobi basis functions, 2024. URL https://arxiv.org/abs/2406.07456.

[2] Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph

Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, April 2024. doi: 10.1145/3620665.3640366. URL `https://pytorch.org/assets/pytorch2-2.pdf`.

[3] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Javier Fernandez-Marques, Yan Gao, Lorenzo Sani, Kwing Hei Li, Titouan Parcollet, Pedro Porto Buarque de Gusmão, and Nicholas D. Lane. Flower: A friendly federated learning research framework, 2022. URL `https://arxiv.org/abs/2007.14390`.

[4] Blealtan. Blealtan/efficient-kan: An efficient pure-pytorch implementation of kolmogorov-arnold network (kan). `https://github.com/Blealtan/efficient-kan`, 2024. GitHub repository.

[5] Alexander Dylan Bodner, Antonio Santiago Tepsich, Jack Natan Spolski, and Santiago Pourteau. Convolutional kolmogorov-arnold networks, 2024. URL `https://arxiv.org/abs/2406.13155`.

[6] Zavareh Bozorgasl and Hao Chen. Wav-kan: Wavelet kolmogorov-arnold networks. *SSRN Electronic Journal*, 2024. ISSN 1556-5068. doi: 10.2139/ssrn.4835325. URL `http://dx.doi.org/10.2139/ssrn.4835325`.

[7] Zavareh Bozorgasl and Hao Chen. An innovative networks in federated learning, 2024. URL `https://arxiv.org/abs/2405.17836`.

[8] Zavareh Bozorgasl and Hao Chen. Wav-kan: Wavelet kolmogorov-arnold networks, 2024. URL `https://arxiv.org/abs/2405.12832`.

[9] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings, 2019. URL `https://arxiv.org/abs/1812.01097`.

[10] Ziheng Cheng, Xinmeng Huang, Pengfei Wu, and Kun Yuan. Momentum benefits non-iid federated learning simply and provably, 2024. URL `https://arxiv.org/abs/2306.16504`.

[11] B. Efron and R. Tibshirani. Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statistical Science*, 1(1):54–75, 1986. ISSN 08834237, 21688745. URL `http://www.jstor.org/stable/2245500`.

[12] R. A. Fisher. Iris. UCI Machine Learning Repository, 1936. DOI: https://doi.org/10.24432/C56C76.

[13] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2018. URL `https://arxiv.org/abs/1608.06993`.

[14] Mehrdad Kiamari, Mohammad Kiamari, and Bhaskar Krishnamachari. Gkan: Graph kolmogorov-arnold networks. *arXiv preprint arXiv:2406.06470*, 2024.

[15] Yann LeCun. The mnist database of handwritten digits. `https://yann.lecun.com/exdb/mnist/`. Accessed: 2024-08-28.

[16] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks, 2020. URL `https://arxiv.org/abs/1812.06127`.

[17] Ziyao Li. Kolmogorov-arnold networks are radial basis function networks, 2024. URL `https://arxiv.org/abs/2405.06721`.

[18] Ziming Liu, Pingchuan Ma, Yixuan Wang, Wojciech Matusik, and Max Tegmark. Kan 2.0: Kolmogorov-arnold networks meet science, 2024. URL `https://arxiv.org/abs/2408.10205`.

[19] 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.

[20] Abdul Majeed and Seong Oun Hwang. A multifaceted survey on federated learning: Fundamentals, paradigm shifts, practical issues, recent developments, partnerships, trade-offs, trustworthiness, and ways forward. *IEEE Access*, 12:84643–84679, 2024. doi: 10.1109/ACCESS.2024.3413069.

[21] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017. URL `https://proceedings.mlr.press/v54/mcmahan17a.html`.

[22] Kleio-Maria Verrou Michail Tsagris, Abdulaziz Alenazi and Nikolaos Pandis. Hypothesis testing for two population means: parametric or non-parametric test? *Journal of Statistical Computation and Simulation*, 90(2):252–270, 2020. doi: 10.1080/00949655.2019.1677659. URL `https://doi.org/10.1080/00949655.2019.1677659`.

[23] Qi Qiu, Tao Zhu, Helin Gong, Liming Chen, and Huansheng Ning. Relu-kan: New kolmogorov-arnold networks that only need matrix addition, dot multiplication, and relu, 2024. URL `https://arxiv.org/abs/2406.02075`.

[24] SpaceLearner. SpaceLearner/JacobiKAN: Kolmogorov-arnold networks (kan) using jacobi polynomials instead of b-splines. `https://github.com/SpaceLearner/JacobiKAN`, 2024. GitHub repository.

[25] Sidharth SS, Keerthana AR, Gokul R, and Anas KP. Chebyshev polynomial-based kolmogorov-arnold networks: An efficient architecture for nonlinear function approximation, 2024. URL `https://arxiv.org/abs/2405.07200`.

[26] Bernard L Welch. The generalization of 'student's' problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947.

[27] Jinfeng Xu, Zheyu Chen, Jinze Li, Shuo Yang, Wei Wang, Xiping Hu, and Edith C. H. Ngai. Fourierkan-gcf: Fourier kolmogorov-arnold network – an effective and efficient feature transformation for graph collaborative filtering, 2024. URL `https://arxiv.org/abs/2406.01034`.

[28] Engin Zeydan, Cristian J. Vaca-Rubio, Luis Blanco, Roberto Pereira, Marius Caus, and Abdullah Aydeger. F-kans: Federated kolmogorov-arnold networks, 2024. URL `https://arxiv.org/abs/2407.20100`.